Universidade de Aveiro
Departamento de Eletrónica, Telecomunicações e Informática
2018

Miguel Filipe Oliveira

Deteção de Propagação de Ameaças e Exfiltração de Dados em Redes Empresariais

Detection of Threats Propagation and Data Exflitration in Corporate Networks

Universidade de Aveiro
2018

Departamento de Eletrónica,
Telecomunicações e Informática

Miguel Filipe Oliveira

Deteção de Propagação de Ameaças e Exfiltração de Dados em Redes Empresariais

Detection of Threats Propagation and Data Exflitration in Corporate Networks

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Professor Doutor Paulo Jorge Salvador Serra Ferreira, Professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president

Prof. Doutor André Ventura da Cruz Marnoto Zúquete
Professor auxiliar da Universidade de Aveiro

vogais / examiners committee

Prof. Doutor Mário João Gonçalves Antunes
Professor adjunto do Instituto Politécnico de Leiria (Arguente)

Prof. Doutor Paulo Jorge Salvador Serra Ferreir
Professor auxiliar da Universidade de Aveiro (Orientador)

agradecimentos /
acknowledgements

Em primeiro lugar, quero agradeçer à minha família. Sem eles não teria o apoio necessário, em todos os sentidos, para concluir este curso. À Joana, pela paciência e cumplicidade que me deu ao longo de todos estes anos, fazendo-me sempre acreditar que tudo é possivel de acontecer desde que lute para tal. Quero também agredecer ao professor Paulo Salvador por toda a orientação e conselhos dados, tanto a nivel académico como profissionais. Um agredecimento geral a todos os meus colegas que estiveram ao meu lado neste percurso, mas um especial reconhecimento para o Fábio Silva pelo seu companheirismo e amizade, que se relevaram essenciais durante os vários meses de dissertação e para o Miguel Bergano, por toda a ajuda e disponibilidade dada.

Palavras Chave

reconhecimento de padrões, análise comportamental, aprendizagem automática, monitorização de rede, classificação da actividade de rede.

Resumo

Nos dias de hoje, várias organizações enfrentam múltiplas ameaças no interior da sua rede. Numa época onde as empresas dependem cada vez mais da informação, estas ameaças podem comprometer seriamente a segurança e a integridade de dados confidenciais. O acesso não autorizado e o uso de programas ilícitos constituem uma forma de penetrar e ultrapassar as barreiras organizacionais, sendo capazes de propagarem-se para outros terminais presentes no interior da rede privada com o intuito de atingir dados confidenciais e segredos comerciais. A eficiência da segurança oferecida pelos sistemas de defesa tradicionais está a ser posta em causa devido ao elevado número de ataques de divulgação de dados sofridos pelas empresas. Desta forma, o desenvolvimento de novos sistemas de monitorização ativos usando inteligência artificial é crucial na medida de atingir uma deteção mais precisa em curtos períodos de tempo. No entanto, a monitorização e o armazenamento dos registos da atividade da rede são restritos e limitados por questões legais e estratégias de privacidade, como a cifra dos dados, visando proteger a confidencialidade das entidades. Esta dissertação propõe metodologias para inferir padrões de comportamento e revelar anomalias através da análise de tráfego que passa na rede, detetando pequenas variações em comparação com o perfil normal de atividade. Delimitado pelas camadas de rede OSI 1 a 4, os dados em bruto são modelados em features, representando observações de rede e, posteriormente, processados por algoritmos de machine learning para classificar a atividade de rede. Assumindo a inevitabilidade de um terminal ser comprometido, este trabalho compreende dois cenários: um ataque que se auto-propaga sobre a rede interna e uma tentativa de exfiltração de dados que envia informações para a rede pública. Embora os processos de criação de features e de modelação tenham sido testados para estes dois casos, é uma operação genérica que pode ser utilizada em cenários mais complexos, bem como em domínios diferentes. O último capítulo inclui uma prova de conceito e descreve o método de criação dos dados, com a utilização de algumas métricas de avaliação de forma a espelhar a performance do modelo. Os testes mostraram resultados promissores, variando entre 96% e 99% para o caso da propagação e entre 86% e 97% relativamente ao roubo de dados.

Keywords

pattern recognition, behavior analysis, machine learning, network monitoring, network activity classification.

Abstract

Modern corporations face nowadays multiple threats within their networks. In an era where companies are tightly dependent on information, these threats can seriously compromise the safety and integrity of sensitive data. Unauthorized access and illicit programs comprise a way of penetrating the corporate networks, able to traversing and propagating to other terminals across the private network, in search of confidential data and business secrets. The efficiency of traditional security defenses are being questioned with the number of data breaches occurred nowadays, being essential the development of new active monitoring systems with artificial intelligence capable to achieve almost perfect detection in very short time frames. However, network monitoring and storage of network activity records are restricted and limited by legal laws and privacy strategies, like encryption, aiming to protect the confidentiality of private parties. This dissertation proposes methodologies to infer behavior patterns and disclose anomalies from network traffic analysis, detecting slight variations compared with the normal profile. Bounded by network OSI layers 1 to 4, raw data are modeled in features, representing network observations, and posteriorly, processed by machine learning algorithms to classify network activity. Assuming the inevitability of a network terminal to be compromised, this work comprises two scenarios: a self-spreading force that propagates over internal network and a data exfiltration charge which dispatch confidential info to the public network. Although features and modeling processes have been tested for these two cases, it is a generic operation that can be used in more complex scenarios as well as in different domains. The last chapter describes the proof of concept scenario and how data was generated, along with some evaluation metrics to perceive the model's performance. The tests manifested promising results, ranging from 96% to 99% for the propagation case and 86% to 97% regarding data exfiltration.

Contents

List of Figures

List of Tables

Glossary

AAA	Authentication, Authorization and Accounting
AI	Artificial Intelligence
APTs	Advanced Persistent Threats
AUC	Area Under the Curve
APs	Access Points
BYOD	Bring your own Device
CIA	Confidentiality, Integrity, Availability
CDF	Cumulative Distribution Function
CHAP	Challenge Handshake Authentication Protocol
CIFS	Common Internet File System
CVE	Common Vulnerabilities and Exposures
DM	Data Mining
DNS	Domain Name System
DMZ	Demilitarized Zone
DHCP	Dynamic Host Configuration Protocol
DPI	Deep Packet Inspection
DT	Decision Trees
DoS	Denial of Service
DDoS	Distributed Denial of Service
EAP	Extensible Authentication Protocol
FCoE	Fiber Channel over Ethernet
FTP	File Transfer Protocol
GRE	Generic Routing Encapsulation
HIDS	Host-based Intrusion Detection Systems
HIPS	Host-based Intrusion Prevention Systems
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IP	Internet Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection Systems
IPS	Intrusion Prevention Systems
IoT	Internet of Things
iSCSI	Internet Small Computer System Interface
IPsec	Internet Security Protocol
ISP	Internet Service Provider
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
L2TP	Layer 2 Tunneling Protocol
MAC	Media Access Control
ML	Machine Learning
MLP	Multi Layer Perceptron
NAS	Network-Attached Storage
NAS	Network Access Server
NSA	National Security Agency
NIDS	Network-based Intrusion Detection Systems
NIPS	Network-based Intrusion Prevention Systems
NN	Neural Network
NFS	Network File System
OSI	Open Systems Interconnection
ORC	Optical Character Recognition
PAP	Password Authentication Protocol
PCA	Principal Component Analysis
PDF	Probability Density Function
PPP	Ponit-to-point Protocol
PPTP	Point–to–Point Tunneling Protocol
RADIUS	Remote Authentication Dial-In User Service
ROC	Receiver Operating Characteristic
Rsync	Remote Sync
SAN	Storage Area Network
SNMP	Simple Network Management Protocol
SCP	Secure Copy Protocol
SMB	Server Message Block
SFTP	Secure Shell (SSH) FIle Transfer Protocol
SSL	Secure Socket Layer
SSH	Secure Shell

SPI	Shallow Packet Inspection
SPAN	Switch Port Analyzer
SVM	Support Vector Machine
TACASC+	Terminal Access Controller Access Control System
TAP	Terminal Access Point
TLS	Transport Layer Security
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
USB	Universal Serial Bus
VLAN	Virtual LAN
VoIP	Voice over Internet Protocol
VPN	Virtual Private Network
WAN	Wide Area Network
WPA2	Wi-Fi Protected Access version 2
WMI	Windows Management Instrumentation
WWW	World Wide Web

CHAPTER 1

Introduction

Companies today have a preponderant factor in the economy, both at the nation and worldwide level, needing a solid structure that supports their expansion, encompassing both architecture implementations and the social sector. The architecture level includes from the physical installations to the complex network environment, where the social field, represents the essence of a company, the workers. Both domains display a hierarchical organization, where the heterogeneous levels of this ordered chain need to cooperate between them. In this way, the existence of measures and policies are crucial to coordinate and guide the diverse elements, either in social as in physical levels, defining the role of each one and directing its actions, for the company well-being. Progressively it is more arduous to supervise so many people, that in big companies, could be dispersed across several geographical points. However, the organization's watchful eye still focuses on the outside of their edges. Not deferring importance to the implementation of exterior defense strategies, employees constitute the main vulnerability of a company (as described in Section 2.2), where most corporate attacks are prompted by insiders [1], which are careless or have a malicious intention. Human factor plays a significant role for the network vulnerabilities, where social engineering comprises a powerful weapon carried by attackers, to target licit users, being those the "open door" for threats, bypass traditional security mechanisms (discussed along of Section 2.1).

Moreover, technology evolution along with Internet proficiency is transforming the way how the world is connected, where digital devices are surfacing, allowing companies to take full advantage and share their services (some of the described in Section 2.1.2) across the globe. As the scope and business expansion, it is essential that companies have the means and mechanisms to ensure the secure access to those services, as discussed in Sections 2.1.3 and 2.1.4. Withal, this evolution not only provides a means for the progress. It also empowers a way to new, more furtive and stealthy attacks that companies need to react and reinforce their security mechanisms to safeguard the safety and integrity of one of most valuable assets: the confidential information. Espionage, performed by nation-state actors or simply by corporation's competitors, aiming to steal industrial secrets and disclose confidential data

related to the employee's records and organization's plans, is a real scenario that hits all types of organizations (being the healthcare, financial and industrial fields more affected). Nowadays, attackers have all the needed resources to break the security walls and spread over the network, stopping only when their objectives are completed. In this way, data breaches are an alarming threat that causes a huge impact on organizations, from the loss of reputation and capital to the disruption of their services. Further, aggressors after violating the network always leave it untouched and without a trace, due to the detection incapacity of the used companies security tools.

In an era where the abundance of data is a fact, and the amount of data that travels in networks is enormous, it becomes increasingly difficult for companies to detect outbreaks in their premature stage, being crucial to adapt and adopt new strategies that increase the security, integrity, and availability of a network. Along these lines, as described in Section 2.5 it is possible to extract meaningful information, and therefore knowledge, from the massive bulk of data using Machine Learning (ML) methodologies (introduced in 2.5.3). This current and highly popular area, with application in the image, speech recognition and detection field, including, for example, fraud and network intrusion detection, came to introduce new solutions that companies need to elevate their network detectability and awareness. However, it is important to notice that, governments are creating more restrained laws to define new standards of privacy rights, accompanied with the technical encryption position, aiming to limit the data gathering action and strengthen the user's privacy.

The work developed in this dissertation introduces a set of methodologies and procedures for detecting anomalous communications in corporate networks. The focus is directed inwards, where spreading threats and data leakage flows to the outside borders are the handled scenarios. ML provide the tools required to reveal subtle patterns of activities and services characteristic of an internal corporate network, identifying and distinguishing their nature between normality and anomaly. Monitoring the environment and gathering the data, without evading the employee's privacy or restricting their freedom, enables the behavior analysis of network activities, so that in the future, such traffic deviations from the normal profile can be discovered in their earlier stage, preventing the propagation to other terminals and close the path to confidential assets.

This dissertation was developed at *IT - Institute of Telecommunications Aveiro* where the workspace and all required resources were kindly provided.

CHAPTER 2

Contextualization and State of Art

Information increasingly assumes a more crucial role for companies in an era where data theft and industrial espionage are recurrent scenarios. New types of attacks are rising and becoming more imperceptible, leading corporations to not despise outbreaks with internal origins and adopt new security strategies. In this chapter, an overview of the security mechanisms and solutions empowered by organizations will be described, along with actual examples that mark the cybersecurity landscape. It will be introduced the ML concept, that provides an essential way to infer meaningful information from the huge amount of monitoring data and plays an imperative role in the anomaly detection area.

2.1 Corporate network security

With network openness and connectivity offered by the Internet, the expansion of sharing data was a key factor for the rise of enterprise business processes. The enterprise networks aptness of providing an easy and spontaneous mean for share and distribute resources is vital and crucial for organizations. Although, it becomes more critical to increase information security. The protection of private and confidential data, for example, financial and commercial information in the transactions, is the primary concern to enterprises. According to [2] there are three major basic information security requirements:

- Confidentiality
- Integrity
- Availability

Confidentiality is the act to ensure that only the appropriate people have access to the information in question. It is ruled by the principle of the least privilege [3], giving users only the necessary features to accomplish the task. At the same time that protects data making it confidential, enacts that only is available to authorized users [4].

Integrity is the capability to only authorized users change sensitive information, that is, ensure that the data contents are correct and are not manipulated from unauthorized people [2].

Availability represents the ability of network information to be readily accessible and usable to the legitimate users. It means that data should always be attainable whenever a legitimate user needs [5].

These points form the basic model of information security, the Confidentiality, Integrity, Availability (CIA) Triad model, as shown in Figure 2.1). The CIA Triad model treats all the three security elements as equal, aiming a data protection scenario where exists absolute confidentiality, perfect integrity and perfect availability. However, such a possibility does not exist in the real world. If a company has a data repository containing commercial information and the security team has designed a system with strong protected rules by using encryption methods and implement permissions that restrict the actions for an unauthorized user. The result is a security plan that has heavy confidentiality and integrity norms, but as this repository is well secure, users cannot access or use the necessary resources. Usually, as you increase the levels of security, it occurs the decrease of the productivity levels. On the other hand, when exists high availability and integrity, many times, the information privacy is put at risk.

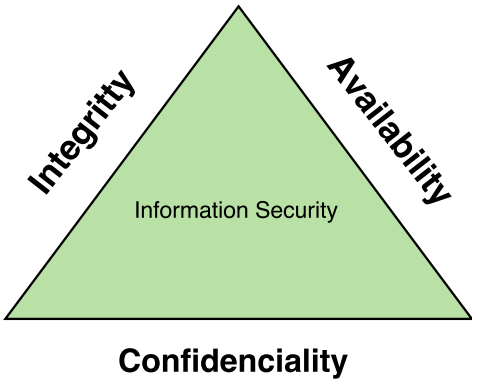

Figure 2.1: CIA triad model

Corporate network security features should protect vital assets. Organizations need to define their critical data and, consequently, elaborate a set of rules and restrictions related to the protection of this organizational assets, the *security policies*. This a high-level document [6] where companies defined the plan to protect and control their network and data, as guidance that tell all employees and business partners the correct way for access and manages corporate network services. This regulation addresses all entities that interact with an organization network, being necessary in order to define what the enterprise users can do. It restricts the applications/software that can be used and installed in their computers, establish the procedures to store and handle with corporate information, and managed the corporation defense aspects, for example, the mandatory use of anti-virus. Also, organizations need to define if personal devices (laptops, tablets, and smartphones) are allowed or if users only can

use the equipments given by the company. This policy, where personal devices are gaining relevance support by the Bring your own Device (BYOD) tendency, requires more secure measures [7]. The Internet of Things (IoT) [8], creates a unified network communication environment, being possible to transfer data between different equipments (like mobile devices, fridges, dishwasher, printers), increasing the number of devices connected to the network. In this way, is crucial to forming policies that control these devices regarding access, traffic, and physical security.

In a network point of view, company administrators must establish security rules on access, defining the authorization and authentication mechanisms to control internal and remote access to the corporate network, as well as, in a wireless communication environment (as will be explained in Section 2.1.3). Also, policies associated to switches, routers and serves [9], along with traffic and monitoring rules should be enacted, e.g., shaping which ports are allowed, what traffic type and protocols can travel on a corporate network and secure approaches to transmit data among networks. In fact, it is common practice for companies to inspect employees' traffic to filter malware and viruses, prevent the leak of intellectual property. However, this traffic analysis is conditioned by established rules limiting the inspection scope (as will be described in Section 2.1.6). On the other hand, to ensure confidentiality and integrity of the classified/private data such as financial or personal records, companies need to define encryption mechanisms in order to secure data within the private enterprise network and in public network (Internet). Data protection is a core security policy that must follow and preserve the encryption rules defined by each country laws [10], being the inspection of encrypted traffic an also severe dilemma.

In such a manner, building a security plan is not trivial. Applying string restrict policies could ensure the desired security, but on the other hand, will limit the user access to information putting obstacles in their work and causing dissatisfaction.

2.1.1 Corporate network architecture

Internet growing contributed to expanding corporation businesses, where people may be in different locations, in several departments and organizational structures around the world. So, corporate networks should ensure that dispersed entities can easily communicate and share resources as if it were a united environment. However, in order to correctly create a communication network, the design at an architectural level must be a careful and well-crafted plan. As follows, this section will cover the relevant concepts the design and implementation of this networks.

The communication network design must adopt the following requirements to accomplish all business objectives [11]:

- The network should be *resilient*, being operational all time, even in the case of failed links and equipment failure. For example, if for some reason, a network fails, even for a second, in financial companies may translate many millions lost. In a hospital case, this may represent lost of lives.

- The network should be *flexible* and *modulate*, being easy to modify and adapt quickly to business change and evolve and support growth and change, correspondingly.
- The network should ensure the delivery of application in reasonable response times between hosts.
- The network should be *secure*, protecting the exchange data and the stored information.

Instead of seen the network as a whole, it is composed by several interconnected components, applying a layered solution to accomplished the requirements described. A hierarchical network structure, where each layer has its own functions, breaks the complex problem of network design into smaller and more manageable areas with fewer devices numbers, being easier to implement and manage the security policies that define the entire security process. As an example, with a non-hierarchical network is to control and monitor the network traffic [12]. Figure 2.2 shows a structure of the hierarchical network model divided into three layers.

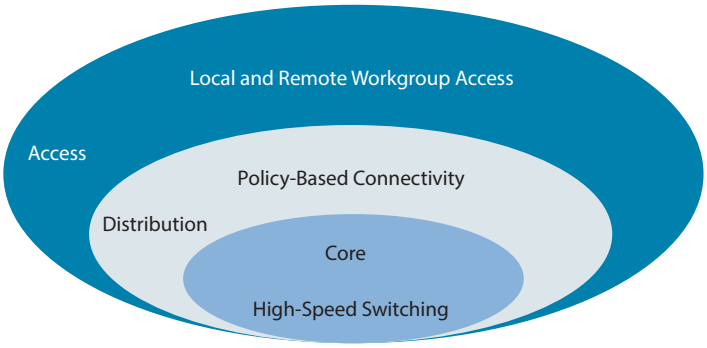

Figure 2.2: Hierarchical network design layers [13]

Access layer provides users access to the enterprise network. Generally, a Local Area Network (LAN) environment [1] grants access to end users, while in Wide Area Network (WAN) (which covers a broader geographic area [14]), the corporate network access is given for remote user and sites [13].

The distribution layer is the intermediate point between the access and core layer. Aggregates the data received from the access layer before it is transmitted to the core layer for routing to its final destination providing policy-based traffic control, where packet inspection can take place [12].

The core layer, also known as backbone network, is designed to route packets as fast as possible. Since this layer must provide maximum availability and reliability with minimum

[1] Type of network that operates over a small physical area where the exchange of information and sharing of resources are straightforward [14]

packer processing, it should not perform any packet manipulation [13]. Note that in a hierarchical model, networks tend to have many single points of failure [11] isolating users from the wanted service. Thus redundant techniques like alternative paths must be applied to overcome these issues. The core layer acts as an integration point to the enterprise edge, data centers, and server farms, where the enterprise edge, usually, contains outside corporate services (as voice, video, and data). In addition, this layer can be linked to the service provider's network, enabling Internet, WAN access and connection to remote sites [13].

To achieve this connectivity the core layer uses routers or multilayer switches, that combine routing and switching in same devices. The distribution layer, in the same way, usually employs Layer 3 devices, such as routers or multilayer switches, in order to enforce access control policies and managing traffic flows [11]. Along with this distribution layer also performs VLAN routing [11]. Access layer devices are located inside a LAN, communicating through Layer 2 switching via a wired infrastructure, that with cables, connects the disjointed buildings. The other way is through wireless access point [11].

With the increase of devices that can attach the network, a usual approach used in corporate environments is VLAN segmentation. It segments the network, enabling the creation of virtual workgroups based on service, users role or location [11]. So, with the formation of sales, human resources and engineering workgroups, for example, along with the VLAN to carry a specified type of traffic (VLAN guest), it is possible to control the access to sensitive data and resources, according to the end users permissions and rights.

Furthermore, companies must delimit the network's boundaries to safeguard the "well-being" of the network and consequently their assets, establishing *zones* [6]. This architectural solution, segments networks, associating interfaces that share common functions or features to which a security policy can be applied. Each zone has an assigned security level based on the level of trust given, setting the security borders of the network, composed of interfaces with identical security requirements. It defines a boundary where the traffic is subject to specific policy restrictions as it crosses to different zones [15]. To support the zoning creation, *firewalls* placed at the network border, acting as a barrier between the different security zones filtering the traffic that flows between them [16]. Typically, organizations divide networks into three zones: the Trust zone referring to the internal network (the private one), the Demilitarized Zone (DMZ) and the Untrusted zone (composed by the Internet). DMZ or "semi-protected" zone [11] has a medium trust level, where can accommodate public services such HyperText Transfer Protocol (HTTP)/ HTTPS, File Transfer Protocol (FTP) and Domain Name System (DNS) [17] allowing to be accessible from the untrusted Internet.

2.1.2 Network services

In order to enable communications over the network, it is necessary to assign an IP address to each device. This can be done automatically incorporating a Dynamic Host Configuration Protocol (DHCP) server, which controls the attribution process and the management of addresses pool. Along with DHCP, DNS, which maps Internet domain names into IP

address [17], providing support and configuration IP services to permit communication in an enterprise network.

Regarding network management, systems administrators can remotely control the state of devices, and handle problems related with performance and issues on the network, through the used of Simple Network Management Protocol (SNMP) [18] by sending informational messages. Similarly, Remote Authentication Dial-In User Service (RADIUS) (explained in more detail in Section 2.1.3), when performing accounting function can be used to diagnose the state of the network [19].

The network offers operational services required to produce work in a corporate such Email [11], World Wide Web (WWW), Voice over Internet Protocol (VoIP), video conference, file and printer sharing. Among the diverse services, this work will give particular emphasis to file sharing.

2.1.2.1 File sharing

With data becoming a most valuable corporate asset, it is essential to provide storage for sharing data over an enterprise network. The idea behind file sharing is that the file remains on the server and is modified by the client, existing only one copy of the file [20]. That is, having a single copy of file that is used on multiple machines. In this manner, in a corporate view, there are two main concepts of data storage on the network: *Storage Area Network (SAN)* and *Network-Attached Storage (NAS)*. In NAS case, storage devices are directly attached to IP networks (LAN), being specially developed to file sharing [13], [21], where multiple workstations can access shared directories and files that are located in a file server. Instead, in SAN, storage devices are not directly connected to LAN. So, when a host needs to access data, primarily interacts with servers, which acts as an interconnection point between workstations and SAN storage devices. Each storage unit is linked to the associated server over some distance [13], and when hosts request files from serves (which have the notion of the file system), the server will provide block-level access to shared data storage using Fibre Channel [13]. Although, for taking advantage of IP networks, communication can be done through Fiber Channel over Ethernet (FCoE) [22], or using Internet Small Computer System Interface (iSCSI) [22] that maps block-oriented storage data over TCP/IP networks using existing IP networks.

On both storage types, typically Network File System (NFS) and Common Internet File System (CIFS) are mostly used file sharing protocols that allowed LAN hosts access resources located on the network. NFS is a standard network file system on all Unix/Linux systems and CIFS, in its turn, is a standard for the Windows operating system; not forgetting of AppleShare for Macintosh. CIFS is a Server Message Block (SMB) based standard that extends all the capabilities of SMB. Most of the time they saw as one once CIFS protocol is a dialect of SMB [23]. Like this, SMB is a Windows network file sharing protocol that enables sharing of files and printers across nodes on the network. It uses a client-server implementation, in that, when a client wants a sharing resource on a network, sends a request message to the server, which will analyze and then forwards a response message. In this way,

when a computer provides a sharing resource with SMB protocol, it becomes a serve [24]. Likewise, to combat the operation system limitation of SMB/CIFS, Samba [25] has become a popular free-software that combine the SMB/CIFS network protocol on Unix systems.

2.1.2.2 File transfer

File transfer empowers another manner to exchange file over the network. Although, dissimilar to file sharing, when a user copies a file to or from a server, one file is kept on the server and another copy of this file remains on the client. Indeed, not existing the notion of multiple hosts handle the same file, in fact, the file "travels" between machines.

FTP is the standard for exchange files on the Internet. It works on a server and client-based architecture not providing encryption rules. In this way, SSH FIle Transfer Protocol (SFTP) was built to create a more secure way of communication. It runs over SSH protocol (responsible for providing a secure communication mean as described in Section 2.1.4), establishing a channel for encrypted transfer information and authentication mechanisms.

For Linux/Unix machines, Secure Copy Protocol (SCP) is another practice for transfer files over the network. On the same way that SFTP, runs on port 22, that is, works with SSH, ensuring confidentiality of the data being transferred and protecting authentication. Nevertheless, in Linux/Unix systems, Remote Sync (Rsync) [20] is a conventional method for remotely copying and synchronizing files and directories, being primarily used on databases synchronization and for administrative purposes, since it allows to transfer just the differences between two sets of files.

Moreover, with WWW popularity, a lot of file transfer is now handled using HTTP. It is a protocol for transmitting hypermedia documents, originally designed for communication between web browsers and web servers, that provides a way for users to upload or download files and pages from the Internet [26]. As will be described in 2.4.1.2, HTTPS is now a tendency for "reform" the old HTTP, endorsing the communication security on the Internet.

2.1.3 Network Access Control

As new network technologies develop rapidly, users have strong demands for network access using terminals, anytime, anywhere. However, traditional enterprise campus networks are IP based networks, manually configured for different office locations, being necessary to apply network access control policies to verify user's access rights. So, to ensure control over the organizational resources, network access control policies are built in authentication - validation of someone or something (user, equipments or network segments) verifying whether what the entity claims to be is, in fact, what and who that relay is [4]; and authorization - establishing what the user should be allowed to do over resources and the user privileges on a system after have access. This constitutes the Authentication, Authorization and Accounting (AAA), implemented by RADIUS and Terminal Access Controller Access Control System (TACASC+) protocols, where all usernames and passwords are keeping in one place, on a central server, enabling remote access control over the network.

So, in AAA architecture, RADIUS servers interact with Network Access Server (NAS), which operates as a RADIUS client, being responsible for sending user information to the server and return the upon response. Thus, when a host wants to enter the network, requesting resource access, initiates Ponit-to-point Protocol (PPP) authentication with NAS, providing the username and password that are sent to RADIUS server. This server supports Password Authentication Protocol (PAP), Challenge Handshake Authentication Protocol (CHAP), Extensible Authentication Protocol (EAP) [27], or alternately consults an external database like Lightweight Directory Access Protocol (LDAP) [28] to authenticate a user by verifying his credentials. With user information retrieval and the configured access policies, the server forwards a message specifying if authorizes the access. In RADIUS, authentication and authorization are coupled together, independently of accounting feature, that could be used for management purposes (2.1.2), indicating the number of resources used and session duration [29].

When clients are connected to LAN through wireless, the IEEE 802.1X standard defines a client-server-based access control and authentication protocol in Wi-Fi environment [30], where each device that wants to attach to a LAN are seen as the *supplicant*. It has a relation with the *authenticator* (such as edge switches or Access Points (APs)), by requesting the identity of a client (workstation) and, posteriorly, deliver the user information to the *authentication server*, usually a RADIUS server. It is responsible for authenticating the user and given, or not, authorization, transmitting the response back to the *authenticator*. To pass the authentication information between the *supplicant* and the *authentication server*, EAP is used and handled by the *authenticator*, acting as an intermediate point (RADIUS client). Besides, Wi-Fi Protected Access version 2 (WPA2) (IEEE 802.11i standard), also provides data protection and access control in wireless LAN connection [31]. Figure 2.3 illustrates the interaction among these components on the authentication process.

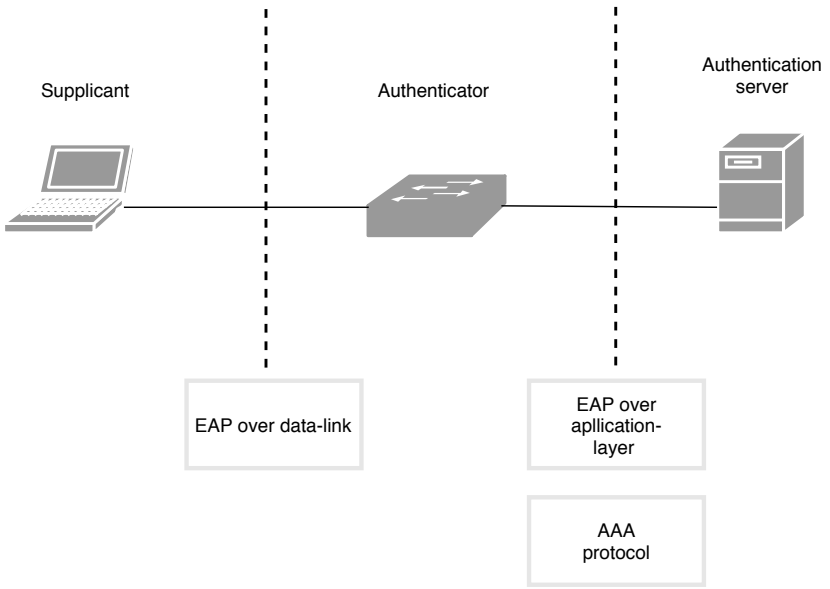

Figure 2.3: IEEE 802.1X Authentication components and process

However, with the IoT and BYOD tendency (as discussed in Section 2.1), network access must be controlled similarly as "normal" workstation. This brings a problem since mobility devices, like printers and phones, may not support 802.1X solution.Thus, to overtake this problem, authentication could be made based in their MAC address. This is called MAC Authentication Bypass (MAB) [32] and should be implemented together with IEEE 802.1X.

According to security polices and user's rights, network devices like switches, can be dynamically assigned by it respective VLAN to grant access. Therefore, unauthorized access is blocked, ensuring the security of enterprise resources and assets, by limiting user's action to a specified VLAN (guest, sysadmin or others according with the policies) like referenced in Section 2.1.1.

2.1.4 Secure Communications Channels

Part of a company's success comes from the ability to communicate between all entities (as employees and business partners), which in case of having a broad scope, they can be distributed by several locations. Moreover, after users have been authorized and authenticated in the network (as described in Section 2.1.3), organizations need to support communications over different types of networks, being necessary to take into account the security challenges. Therefore corporations deploy secure communications channels over an unprotected network, like the Internet, providing three essential data security requirements: origin authentication, data integrity (2.1) and data encryption to ensure confidentiality (2.1) [11], [13].

The concept "Office from home" [33], where business extends all it is internal resources to employees that can access them from their homes, is an actual scenario, which corporations address by employing Virtual Private Network (VPN) solution. It is a private network that uses a public network to connect users and remotes sites, being essential to ensure the security data points above described over an unprotected network. VPN can be of two types: *remote-access* and *site-to-site*. *Remote-access* VPN, enables users to establish a secure connection from their homes and stations, to the corporate network through the **isps!** (**isps!**) [16]. This is formed by several points of presence (POP), providing access to resources (according to their permissions), as if they directly plugged to the server. It is enforced using several protocols: Point–to–Point Tunneling Protocol (PPTP), Layer 2 Tunneling Protocol (L2TP) over Internet Security Protocol (IPsec) [33], Secure Socket Layer (SSL) or Transport Layer Security (TLS) [13] and SSH [17].

On the other hand, *site-to-site* VPN, secure connections between main enterprise and remote office branches [13], allows stations in multiple locations to exchange secure information over the public network (Internet). In this VPN type, the most common secure tunneling protocol is the IPsec, with some variants like static or dynamic configuration and implementation of traffic protection using Generic Routing Encapsulation (GRE) [11]. In both VPN approaches, IPsec is suitable when two participants want to communicate ensuring authentication and data encapsulation (encryption). IPsec has two modes, tunnel or transport [11], according to the way they were implemented: on the ingress router or in the end host, respectively.

2.1.5 Policy Zone-Based

With the creation of zones (concept elucidated in Section 2.1.1), corporation assets will be defined by similar security requirements that establish their network borders. Thus, traffic can flow between different zones, although, cannot be free. For this reason, paths between zones should establish rules to control traffic that is traveling through them. Firewalls bind the path between zones, acting as "check points", in which, the packets that travel between security zones are confronted with the security policies implemented, filtering the packets that obey to the corporation policy, from other ones, that should be dropped.

Packet inspection and filtering are one of most fundamental functionalities of a firewall. It is controlled by security policies, that supervises the inbound traffic based on source and destination, preventing connections from an unprotected network (Internet) to enter the private LAN. If a web server is placed on a trusted zone, administrators, for example, only permit traffic from Internet HTTP native from port 80. Yet, security policies apply to outbound traffic, preventing employees from accessing forbidden content and places on the Internet that are in a blacklist. On the other hand, it establishes rules for tracking the IP addresses that have the authorization to connect with outside networks. This intermediate points can also control bandwidth usage per user or desktop, according to the policy.

These filtering capabilities performed by a firewall are conditioned by the type of packet inspection. It can filter all the packets performed by a statically set of rules, that looks inside each packet, checking the source/destination addresses and ports (and sometimes protocol), but, without connection notion [16] - *stateless* filtering. More recent firewall generation supported by most vendors, Check Point [34], Cisco [16] and Juniper Networks [35], perform a *stateful* packet filtering, which incorporates the concept of connection and state [6]. It treats all packets belonging to one connection as a data flow, instead of isolated entities as in *stateless* filtering, where a session is dynamically created for the first packet, and, the firewall will allow data back based in that connection.

2.1.6 Monitoring

At a corporate level, the concept of monitoring usually considers logging entrance to surveillance systems, to prevent entry of unwanted entities. The same happens on networks, but not so direct and straightforward. Audit and supervise the network well-being, requires the analysis and, subsequent, understanding of network traffic.

Due to today's attacks [36], firewalls cannot give full access control over the network, since this threats can overtake the security policies by misleads of the inspection mechanisms (described in Section 2.1.5), as the case of Internet Protocol (IP) and Media Access Control (MAC) spoofing or the smurf attack. So, another level of security is needed, gathering data and detecting intruder's activities that violate the established rules.

Intrusion Detection Systems (IDS) is seen as a primary defense along with the firewalls [37], exposing unauthorized and harmful activities, being able to act at host level (Host-based Intrusion Detection Systems (HIDS)) [38], or the network level (Network-based Intrusion

Detection Systems (NIDS)). The later monitors the network traffic that can be a system threat [37], reading all incoming packets and trying to find any suspicious pattern. It spots intrusion activities based on a set of rules or signatures of known attacks. This exposure mechanism is known as *Misuse-based* or *Signature-based* detection. Although, when faced with new malevolent forms, unknown by the system, it will never detect them [39]. Once each intrusive activity is uniquely represented by a pattern or a signature, this mechanism is efficient in detecting recognized attacks [17]. However, because of the time that the signature databases had until patch the new pattern, it is not efficient for unidentified attacks [37].

On the other hand, an *Anomaly-based* detection mechanism can be more adjustable in a way that it can learn with the malevolent intruder forms. It builds a profile with normal activity and checks for deviations with the natural profile, using ML (explained in Section 2.5.3), statistical and knowledge-based techniques (as discussed in Section 2.5) [37], [40]. Based on the behavior of the attacks, and dissimilar to *Signature-based* monitoring, it can detect new pattern activities that do not know, but has a crucial constraint: it assumes that all intrusive actions are necessarily anomalous [41], leading to the occurrence of false positives [2] [42].

The control of unauthorized access and the maintenance of the confidentiality, integrity, and availability of information (2.1), is not complete by only using a security system with IDS [43]. After being noticed, it is necessary to act and take several measures to stop the attack from spreading and reach the terminal points where valued assets are. This passive mode does not eliminate the threat. It only stores the log audits and informed administrators, who has the responsibility to decide how to deal with those signs [43].

In contrast, Intrusion Prevention Systems (IPS) also monitors the traffic and alerts when it discovers some anomaly, but takes a proactive action by dropping the malicious packets and block the illegal traffic. The goal is to stop attacks that reach a host or a private network [44], at the same time that logs the malicious data [45]. Like IDS, the prevention can be divided into Host-based Intrusion Prevention Systems (HIPS) (where prevents the entry of attack into an individual host) and Network-based Intrusion Prevention Systems (NIPS) (prevents and identifies attacks at the entry point) [14]. So, an IPS typically is an IDS that can detect and prevent attacks [11], combining firewall capabilities for selecting acceptable traffic based on defined rules, and inspecting capabilities by probing the packets sent from the firewall [6], to stop attacks in real time [44].

IPS and IDS go beyond the standard packet inspection (also known as SPI), which examines both source and destination address, port number and, sometimes goes deeply, inspecting also protocols (Transmission Control Protocol (TCP) and User Datagram Protocol (UDP)) [16], [46], limited by the transport layer of Open Systems Interconnection (OSI) model (illustrated in Figure 2.4). The goal of a more in-depth inspection is to deny the packets that bypass the firewalls and carry malicious content that goes unnoticed. DPI provides application inspection, looking in detail at the contents of data payload that are being sent, operating at Layer 7 of OSI model (Figure 2.4). However, the deepness of monitoring can also be an issue and rise

[2] Anomalous activities that are not intrusive but are flagged as one

legal concerns. For example, Internet Service Provider (ISP) can monitoring network traffic to find malicious virus, yet, many times, the personal information data is sold to marketing and advertising companies [47]. Between those two extremes is a 'grey area' where the privacy violation is an intensely debated point [48], [49], being encryption of data a vital concern to corporations, and a rising tendency [50], which aims to protect data privacy, but at the same time, becoming harder to detect spiteful activity directly [51].

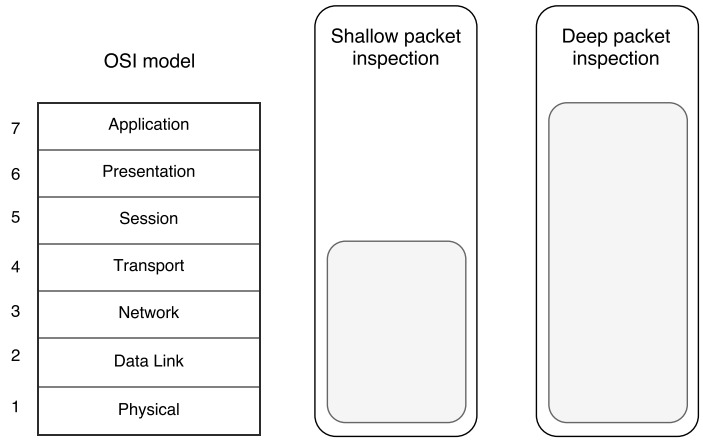

Figure 2.4: Comparison between scope of SPI and DPI on OSI model

Concentrating on intrusion from outside the networks is a good practice to prevent unwanted accesses, but at the same time, internal network activities should not be ignored (as will see in more detail in Section 2.2). Nearly every computing devices have the capability of producing logs which can express, for example, authentication records or failed password attempts. So, these audit messages must be collected and stored in appropriate places like remote servers for security and analysis purposes. Monitoring and reviewing audit records permit companies spot and prevent a breach in its premature stage. Also, it is common that network and system administrators use log records, even after some hours after, to understand what has happened and fix the problem source. However, the larger the organization, the more logs there are what is a significant obstacle concerning management. In this way, instead, IPS that only see IP addresses, packets and protocols, empower a log monitoring solution can give a forensic analysis over the network and an event correlation.

2.2 Corporate network threats

Even when organizations have the most protected and elaborate security system, there will always be an "open door" or a mistake that will cause vulnerability for attackers to take advantage off, in the way that perfect security never can be reached [17].

A *threat* or network attack occurs from an attacker who seeks to find and exploit vulnerabilities, trying to harm a system or disrupt normal business operations. So, *vulnerabilities* are defined as weak points or loopholes in security that an aggressor exploits in order to gain

network access or to reach resources on the network. Thereby, a threat may exist, but if there are no vulnerabilities for the threat to exploit, then there would be no risk [2].

It is important to distinguish who or what and the goal of these threats that outbreaks the enterprise networks. Attackers can be seen as unauthorized users of the system that cracks the secure barrier - external attackers. Whereas, internal intruders are those that have permission to access the system [14].

The first type that constitutes the external threats is mainly represented by criminal hackers, hacktivists, nation-state actors (attacks sponsored by governments) [2] or enterprise competitors. Intruders are no longer motivated by notoriety. Instead, economic or political gain and government/industrial espionage are the actual causes [9]. This shift enables the creation of new malicious forms making intruders always one step ahead. It also came to alter the mode completely how they act, as said by Heidi Shey, Senior Analyst at Forester, "Hackers are carefully picking their victim organization, learning its businesses, understanding its partner relationships, and testing for weaknesses and vulnerabilities" [52]. New methods are continually emerging in volume, variety, and velocity [53], exploring unknown and undiscovered vulnerabilities. Due to this fact, and because there is a discrepancy between what an intruder knows and what the defender recognize to defend against these threats [54], are known as *zero-day attacks* and are one most challenging tasks to enterprise network security [55]. To emphasize, public identified vulnerabilities are listed in Common Vulnerabilities and Exposures (CVE) helping companies to mitigate the known weak points [56]. Furthermore, the actual cyber intrusions are becoming stealthier and persistent [57] using more sophisticated tools. Invaders are opting for targeted attack mode [58], [59], as shown in Figure 2.5, comparing to the traditional attacks methods.

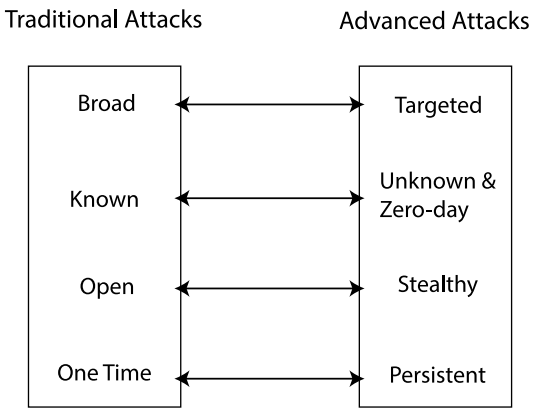

Figure 2.5: Traditional vs. advanced attacks

After differentiating the external from internal attackers, was given a small enhancement to the internal threats, which is usually the basic behavior of almost all companies [60]. The

traditional network architectures were built by placing the jewels of the crown (the company assets) in a well-guarded castle. Organizations think that a secure perimeter protects all critical resources and nothing can pass. What if an unauthorized entity is already inside the castle?

Old fashion cybersecurity thoughts, that direct the focus of the defense to outside attacks, keep assuming that the danger only has external origins [61]. It also claims that the outside network, unlike the private network, is the one that is unprotected and a danger for corporations. This differentiation, where everything within the boundaries of an organization is considered to be reliable, and everything else (such as the Internet) is considered as untrusted, and in today's world, it is not entirely true.

According to the Clearswift Insider Threat Index (CITI) annual report 2017 [1], 74% of security incidents came from the extended enterprise, demonstrating that, nowadays, insider threats are a huge risk for companies. Insider users having permissions to access the organizational system, which can be considerate an employee, contractor or other trusted third party [60]. However, according to the privileges, they are divided into two groups, malicious insider (or intentional insider) and accidental (or non-malicious insider) [60], [61].

The first type is an insider that uses their privileges intentionally to compromise the corporate network, system, and data, affecting the confidentiality, integrity, and availability (2.1) [62] or physical well-being of the organization's information [63]. Usually, their motivations are personal or financial gain, espionage or a mean for revenge [60]. As an example of malicious insider is the case of Edward Snowden, National Security Agency (NSA) contractor who stole information and provided it to the media, as well as, likely to Russia and China [64].

The other type, are within users that involuntary and accidentally can cause damage and, by mistake without bad intent, open breaches in the system, compromising the confidential enterprise data [65], [66]. Currently, these non-malicious users are considered as the most abundant threat in an organization ecosystem, as the Ponemon Institute shows in 2016 Cost of Insider Threats Study [67], that among 874 incidents reported by companies 568 were caused by an employee or contractor negligence. Due to this fact, employees comprising a vulnerability source to the company's security, empowering attack vectors for malicious insiders and external adversaries gaining system access [63]. Outsider attackers do not have authorized permissions, so, they maneuver the insider users as "puppets" to gain entry and to accomplish their purposes. In insider-threat literature [65], [68], this type is classified as *masquerade insider*.

2.3 Attack objectives and methodology

Once an intruder causes a breach in the system, one of most severe threats to companies is the data theft or leakage, formally referred as *data exfiltration* [69]. As defined in [70], is an illegal transport of data from within an organization to an external destination, that may cause substantial financial and reputation damage by disclosing trade secrets, project futures plans, and national secrets if at governmental level [69]. A significant data leakage

is the attack to Equifax, being one of the biggest data breacher ever notice [71] remaining undiscovered for weeks, where 143 million American's personal information was stolen [72], along with Target Corporation's incident (2013) [73], where a massive amount of credit and debit card numbers were compromised. In data breach's sequence, film companies in recent years have been the primary target, from Sony's data breach [74] to the famous ransom attack to HBO's company [75], where the attackers stole multiple episodes of Game of Thrones show, demanding to HBO payment in order to not release the upcoming episodes. Most of times information of this attacks is restricted and not disclosed by the attacked companies.

Not all leakage is related to data/information. Resource leakages, where attackers effectively are not stealing anything more than computer power from their victims (CPU power and its requisite electricity) for Cryptomining (cryptocurrency mining) [76] [3], could cause an impact on business operations [77]. In the same way, spam campaigns could also use the computer's resources to broadcast without the consent of the owner.

Still, not all attacks are stealing-based. Not being the focus of this work, Denial of Service (DoS) is responsible for causing service disruption making company's services unavailable to its legitimate users. These attacks interfere with the availability component that corporations must ensure (2.1), where reputation impact and loss of market are the consequences that can be extremely costly. Driven by the increasingly large number of vulnerable systems, attackers could launch a Distributed Denial of Service (DDoS), which uses a large number of computers to coordinated a DoS attack against a single machine or multiple victim machines [78]. Mirai's botnet caused disruption of the Internet targeting Dyn company, one of the biggest DNS providers (service introduced in Section 2.1.2) [79].

The year of 2017 was the ransomware's year. It was verified a colossal increasing grow of ransomware as a computer crime hitting all types of organizations [80]. It all starts with the famous and fiercely WannaCry (also known as WannaCrypt) attack that shut down computers all over the world, as Figure 2.6 illustrates, changing the way of ransom acts and how it was seen, becoming the first "ransomworm" ever seen. This malware, blocks access to a computer or its data, using encryption techniques and demands money (generally in Bitcoins) to recover it. This is the behavior of common ransomware. The worm capability is what makes it unique and allows it to spread around the Internet. Wannacry took advantaged of Eternalblue, one of the several NSA exploitation tools (such as EternalChampion, EternalRomance, Eternalsynergy, and so on) leaked by an anonymous group calling itself Shadow Brokers [81]. These tools are based on multiples vulnerabilities in the Windows implementation of SMB protocol and allowed this worm propagate through the network using SMBv1/SMBv2 vulnerabilities. It is important salient the spread action mode of this worm. It checks the IP address of the infected machine and attempts to connect to TCP port 445 (SMB) of each host in the same subnet and also spread to the Internet by generating a random IP address. When the incursion gains access and starts to run in the machine, it contacts a domain, the killswitch (i.e., event that is used to stop a program from continuing to execute) domain found by accident. It attempts an HTTP GET request to the domain: if it succeeded, the intrusion stops, otherwise,

[3] Currently, the most valuable currency to mine with standard systems is Monero

the incursion continues [81], [82]. Besides, Wannacry utilized a Doublepulsar backdoor for established persistence and installed the ransomware payload in the infected machine (the malware executable that encrypts the files).

The exposure of NSA exploits and hacking tools, not only caused a significant impact across several corporations such the US delivery company FedEx, the Spanish Telefonica and several UK hospitals [83] (relative to WannaCry's impact), but opened the way to new attacks becoming more "virulent" using the vulnerability in SMB protocol (patched by Microsoft, MS17-010 [84]) for spreading, like Bad Rabbit [85], NotPetya [86]; and more recently RedisWannaMine [87].

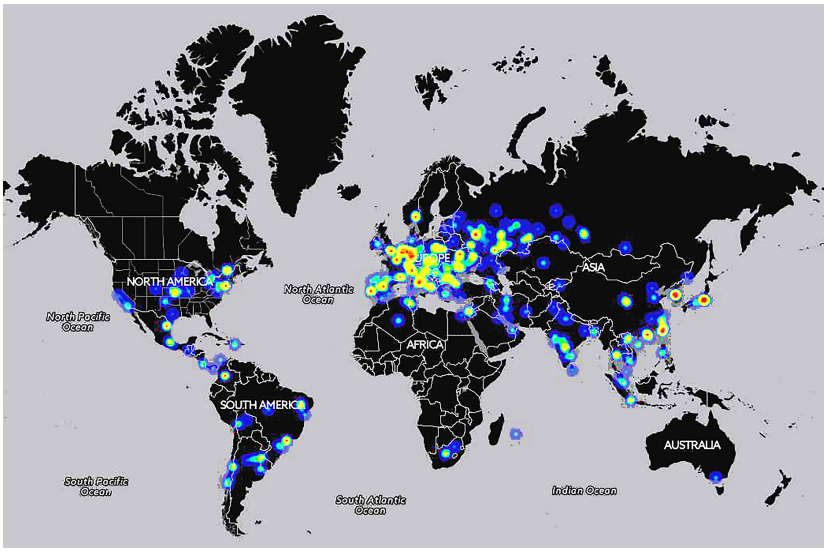

Figure 2.6: Heatmap of the areas most affected by Wannacry attack 11:07 AM, 12 May 2017 [88]

NotPetya is a complementary version of Wannacry. Both uses Eternablue exploit for spreading and encrypt files demanding bitcoins to recover them, and in NotPetya, other leaked exploitation was used: EternalRomance. However, it does not depend exclusively on SMBv1 vulnerably for spreading. If the target was already patched with MS17-010 vulnerability, Notpetya ransomware empowers password harvesting tools (like the open source Mimikatz), that are delivered and used by PsExec and Windows Management Instrumentation (WMI), both Window's legitimate tools, for lateral movement infecting new machines on the same LAN [89]. Therefore, this improved version was designed only to spread internally (in a LAN) and not on the Internet, like Wannacry was. The initial infection was using a software update for M.E.Doc [4], where attackers wanted to contaminate every foreign organization that did businesses with Ukraine. Thus, seems that Wannacry was a a destructive action, once initially infecting small amount of machines, but spreading in world scale, whereas NotPetya at the beginning infected a vast network but was limited to the internal corporate network confines [90].

The cyber security landscape has changed. Compared with traditional attacks, new exploits are designed to have a global impact within minutes, using persistent and stealthy

[4]Popular accounting software used by Ukrainian companies

actions, as demonstrated in Figure 2.5. Based on reported incidents [91]–[93], many high-profile organizations have experienced a new kind of cyber attacks, Advanced Persistent Threats (APTs). These cyber attacks aim to penetrate a system, slowly, and without causing any harm at first, but always with the final goal in mind: strike and becoming too late for the infected system to recover or prevent the attack. The INFOSEC Institute estimates that advanced cyber attacks can stay inside a network for more than 200 days on average before being discovered [94]. Therefore, this section will describe the several phases which characterize these serious threats.

2.3.1 Knowledge acquisition

Before launching the intrusion, attackers study the vulnerabilities of their target. Since it can be anyone in an organization, from employees to administrators, invaders search and gather useful information about them. Social engineering as defined in [95] is "one of the simplest methods to gather information about a target through the process of exploiting human weakness that is inherited to every organization", in order to pose a trusted insider. Performed by *masquerade insiders* (a concept discussed in Section 2.2), external attackers take advantage of company's users that have authorization and permissions. They employ several simple techniques from direct knowledge of the company's secure facilities and organizational structure to simple listen employee's conversations. Although, such direct approaches sometimes do not work. So, more elaborate methods like crawling company's websites, mailing lists, social networks, and *dumpster diving*, are used in a search for valuable information such as personal data and valid credentials [96].

Once the target is identified and studied, the next step is to find a flaw that allows the attackers to gain access. This is usually accomplished by scanning an organization's network and environment to find security holes and "open doors" granting a way for the attacker to come in.

2.3.2 Infiltration

When the target network weaknesses are known, intruders, start looking for an initial exploitation to find a way to get inside the network. Using the information collected in the previous phase (2.3.1), and once again, applying social engineering tricks to luring insider users, taking advantage of their goodness and ingenuousness, empowers a way for intruders gain access. *Phishing* is the most common and effective attack vector, where "normal" emails or instant messages, containing most of the time, malware attached or embed links to redirect users to suspicious websites [97]. Phishing exploits the curiosity and empathy of users, gaining their trust through faking the identity of higher entities such as enterprise superiors, bank and judicial personalities (impersonation); or, on the other hand, by intimidating, leading them to open these poisoned attachments. With the increasing popularity of social networks, phishing attacks are changing their delivered methods, using more sophisticated techniques targeting particular individuals or groups [96], rather than the old fashioned spam way, the

Spear-phishing. Also, watering hole attacks [98], that fabricate websites, stealthily redirect users to another site, infecting the machine with injected malware. The malicious software that compromises the user's machines and gives the attacker a foot in the door can be simple trojans, rootkits or keyloggers. This first occurrence of malware does not necessarily cause any damage by itself. Its purpose is to infect further machines, and to wait for the remotely triggered deploy, implementing the Propagation phase, that will be elucidated in Section 2.3.4. Notice that, the malware samples could exploit known CVE, or, contrary, *zero-day* vulnerabilities.

Other straightforward ways, including deceiving the enterprise concierge, to gain access into the installations, by searching post-it notes (where users wrote there passwords) and unlock computer's, to simple infection via contaminated hardware. *Baiting* is a technique that mines the user's curiosity putting, for example, top-secret labels on stored devices that are infected with some malware [96] or placing them in a tempting place. So, users, in analogy to mouses, falls into the trap and yielded to temptation, opening the content of this devices.

In major Target data breach, discussed in Section 2.3, according to [73], attackers gain access into Target's network, exploiting the weak point found on Fazio Mechanical Services System [5], using a trojan initially installed through a phishing attempt. In similar lines, Stuart McClure, CEO of computer security firm Cylance, in an interview for the POLITICO [99], affirmed that in Sony's attack (2.3) hackers got inside across phishing emails with a forging link of Apple website asking for logon ID and password of staff. On the hopes that people were using the same password for work and personal accounts, posteriorly, attackers searched into LinkedIn for the victim's profiles to deduced Sony network logon IDs.

2.3.3 Learning

The initial onset of malware delivery (2.3.2) is to:

- Establish a foothold into environments that the attackers do not control.
- Gain access to an individual machine on network.
- In case the main target has not yet been reached, obtain information of such network hierarchy and segmentation 2.1.1, the running services 2.1.2 or internal servers accessed by the compromised computer to staying undetected over prolonged periods and accomplish their goals.

Following the Target breach analysis [73], after compromise Fazio Mechanical and found a door into the Target's network, attackers perform a kind of inside knowledge acquisition, in order to understand the network and determine which systems need to be compromised [2]. This learning process will help the attacker to map the network and acquire intelligence about their next move. Network tools to find active connections, open ports (e.g., Netstat [100]) and scanning the network (e.g., Nmap [101]) are readily available to intruders. Furthermore, it can also steal login credentials from computers that deal with valuable information, normally performed by the company's administrators, power the attacker's movement across the network.

[5] Third party vendor, which is a heating, ventilation, and air-conditioning firm

Notice that, social engineering methods, once again, can be used to obtain this necessary credentials.

2.3.4 Propagation

After the Learning phase (2.3.3), attackers will have all they need to escalate privileges to an admin level and start infecting others machines in order to reach the end target. This process of expansion is commonly known as *lateral movement* [102], [103] where intruders progressively spread through a network looking for high-value data inside the network. One of the final goals of this work is to recognize and posteriorly prevent/stop this contagious move. The propagation behavior marks the action mode of the today's attacks, where the goal is to infect the most significant number of machines, but, at the same time, remaining discrete and unknown for as long as possible. WannaCry and NotPetya ransomware (described in Section 2.3) spread mode shows the magnitude of propagation across the entire world, staying unstoppable for some time. Also, the cyber espionage Stuxnet's worm, that invaded Iran's nuclear facilities, infected more than 50.000 Windows computers, even if they were offline [104]. The initial infection may have been caused by a hand-held Universal Serial Bus (USB) device that carried the attack code [93], [104]. Stuxnet did not spread indiscriminately. It identified a specific target to take control of centrifuges, erasing the track, by providing false feedback to outside controllers [105]. This complex action mode allows to be undetected over a ten months [93].

Most of the times, propagation is associated with the establishment of persistence [102], in which threat actors create additional points granting a way to remotely execute instructions and install malware over command and control channels (C&C), using remote access tool (RAT) [59] or backdoors [65]. With a stealthy and undercover spreading, attackers can travel among the network and obtain access to the desired target, where are the valuable assets.

2.3.5 Exfiltration

When aggressors finally reach the goal, it is time to send, confidential or secret information from an infected or attacker-controlled target system, to outside of the company's network where the attacker's machine is. The identification of data exfiltration action also is one of the primary goals of this dissertation, namely, consider the disability of security and monitor mechanisms to detect this illegal data transport. The effectiveness of the exfiltration lies in the attacker's ability not to leave a footprint, remaining undetectable. It can be challenging for monitor systems to discern the traffic that carries the sensitive data from the normal traffic, once attackers take advantage of the increased encrypted traffic used in communications as described in Section 2.1.4.

To compress and mask the exfiltrated traffic, a channel between the infected and intruder machine is created, where the aggressor, within the several existing possibilities, lays on data transfer protocols, e.g., FTP, HTTP (and the secure version using SSL, i.e., HTTPS), DNS and email [106] to exchange the information. With steganography techniques, it is possible to

hide information within images, audio, video files [107] becoming more difficult to detect, as happened in Shamoon attack to Saudi Arabia oil producer's, where user's credentials were hidden in a JPEG image [108].

2.3.5.1 Aggregation

The exfiltration's success depends on the intruder's obscure capacity. To make detection more arduous, the covert channel, may not transfer sensitive data to the external network directly. The exfiltration process could be a multi-phase stage, composed by more than one hop, where at some point, the data collected is aggregated and sent to the final destination. This is what happened on Target's attack [73], [109]. The infected terminals sent data to a compromised internal staging server, where it was stored and aggregated until it was a suitable time to transmit to the attacker. Later, was found that the credit card information, was sent in bulk to a compromised FTP server belonging to a third party and finally pushed to legitimate sites.

2.4 Monitoring

The increase of volume traffic along with the attacks escalation and diversity, rises company's concerns being network monitoring a key point to managing an environment. In this way network monitoring and information security walk side by side. However, with the increasing of network data volume, it is unreasonable and unpractical for network sensors to collect all traffic that travels in the environment. Specifying and filtering the meaningful information becomes vital for better auditing, and posteriorly, being easier to analyze the acquired data and not overload the storage units.

Traffic volume data is noisy. It contains packets that carry non-relevant information and can be ignored. Despite existing several network protocols, the majority of traffic is originated from the TCP, UDP and Internet Control Message Protocol (ICMP) protocols [110].

2.4.1 Active vs Passive monitoring

The monitoring of network health requires data acquisition, and it is essential to distinguish the two primary techniques: *active* and *passive* monitoring. Passive mode analyzes the network traffic for a specific time and reports the results. Active monitoring, implants test packets to measure the performance and availability of network services, i.e., how fast the packets travel through the network. Active monitoring relies on the injecting capacity of test packets, generating data to evaluate particular aspects of network performance. Since it introduces extra traffic in the environment, it performs real-time monitoring. On the other hand, passive monitoring, audits historic data, providing a performance overview of the whole network.

Both approaches have their strengths and drawbacks. Passive mode collects a larger volume of heterogeneous data and can cause overhead to networking hardware. Instead, active monitoring gathers smaller amounts of data and can act inside or outside of the network

environment, in place of passive monitoring, that only measures internal traffic, limited to the network that the device is connected. Thus, active auditing is ideal for measure a specific network metric, ensuring good Quality of Service (QoS) practices, and passive examination is useful to handle large data volumes, capable of establishing bandwidth and traffic standards (based on the usage history). Considering that it analyzes data for a long time, it can more easily profile the network traffic identifying potential security threats [111].

2.4.1.1 Active Monitoring

Active tests detect real-time network and application problems like end-to-end reachability, packet loss, and jitter. Some examples of active monitoring tools and techniques will be described below [112]:

- SNMP

  As referenced in 2.1.2, SNMP permits to manage the network performance finding and solve problems. This protocol had three main components: *Managed devices*, *Agents* and *Network Management System*. The managed devices are network nodes (such as routers, switches, printers, and so one) that support SNMP and integrate the agent. Further, the agent is a software that can access and deal with the management information, acting as interpreter between the managed device and the Network Management System, that coordinates all the managed devices through SNMP.
- ICMP

  This protocol can diagnose and determine the reachability and availability of a network device likewise as it is also used to infer the delay and loss of packets.
- Syslog [20]

  Each device (machines, routers, hubs, etc.) is continuously confronted with a logging system message that will notify with a response. The logging data is stored in a central repository reporting system status and usage information.

In common all the describe methods generate some input in the network using it to invoke information or perform some test. Although, some additional tools merely generate extra traffic to measure the network state:

- Ping and traceroute

  Useful to estimate delay time and packets loss. Both check the reachability of IP hosts through ICMP packets.
- Iperf

  A software tool which measures the maximum achievable bandwidth in the network, as well as, packet loss and delay jitter [113].

2.4.1.2 Passive Monitoring

Passive monitoring does not introduce extra packets in the environment, instead only collects the traffic that travels in the network during a particular time. There are two approaches to extract the travel data: Switch Port Analyzer (SPAN) and network TAP.

SPAN, alias *port mirroring*, copies packets flowing among different ports and entering VLAN to a common interface that can be placed on a remote host - the mirror port. Usually, it is implemented in switches (as well as in routers) not being required additional hardware, and the duplicate traffic posteriorly is analyzed to enforcing policies, detecting intrusions, monitoring and predicting traffic patterns. Relatively to performance questions, mirror traffic can be from multiple ports, exceeding the mirror interface capacity (resulting in dropped packets). A typical procedure to avoid this degrading factor is limiting the number of interfaces, and VLAN monitored. Another and more robust solution is to incorporate a firewall to filter traffic sending only the necessary packets to mirror [114].

Network TAP also mirrors traffic but using a dedicated hardware device placed on a cable infrastructure between network devices (such as firewalls, routers, and switches), as shown in Figure 2.7. Instead of copying the traffic data to a specific port, it duplicates packets directly on cables. The TAP device should be placed in secure locations and can be of two types: passive or active. Passive TAP does not consumes its power capacity and does not interacts directly with other network elements. It implements an internal splitter mechanism, supported by switch ports where network traffic flows; and monitoring ports, which are used to transmit the signals of duplicate packets. This separation does not interfere with original data that flows on the network, safeguarding that every each packet is mirrored and eliminates the possibility of oversubscription (which can happen in SPAN). Oppositely, active TAP, upon receiving a message, it instantly retransmits traffic for the network and monitoring destinations (without split criterion) and requires their power source to perform the task. Indeed, if a power outage happens, active TAP represents a failure network point (although advanced techniques already handle with this power problem) [115].

Concluding, although TAP presents better arguments about the port mirror, the choice of the monitoring solution depends on the situation in question. TAP are expensive equipments that drive companies to spend extra money, not forgetting of the inconvenience when opening and moving cables during installation or re-installation. On the other hand, port mirror does not introduce extra monetary resources, and it only depends on the switch port capacity. So, in the scope of this work, monitoring of the network welfare is a port mirroring solution, once switches have the responsibility to forward the packets to their destinations, and at the same time, could control the traffic of a single and required VLAN.

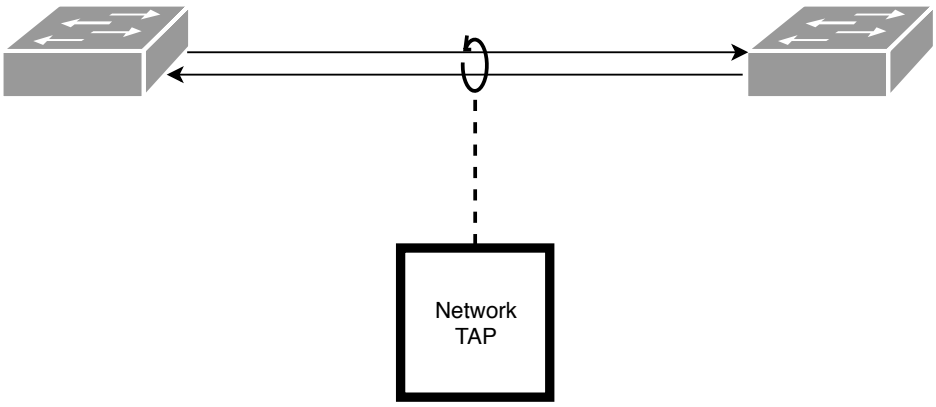

Figure 2.7: Network TAP settled in direct cables between two switches. All the traffic that travels through the cables are duplicated into the TAP device.

After the data has been collected from various point within the network, it is time for traffic analysis on the data. A packet by packet is impractical due to the system's latency and the amount of both software and hardware needed. Instead, a statistical approach is a good starting point to extract information (an approach that will be discussed in Section 3.2). Although, it is essential to always have in mind the OSI model (Figure 2.4). As you move higher up into the model, traffic sniffing becomes more complex, and the consequent packet inspection rises legal issues [116]. Currently, the network has been experienced the increase in average volume of encrypted Internet traffic as illustrated in Figure 2.8.

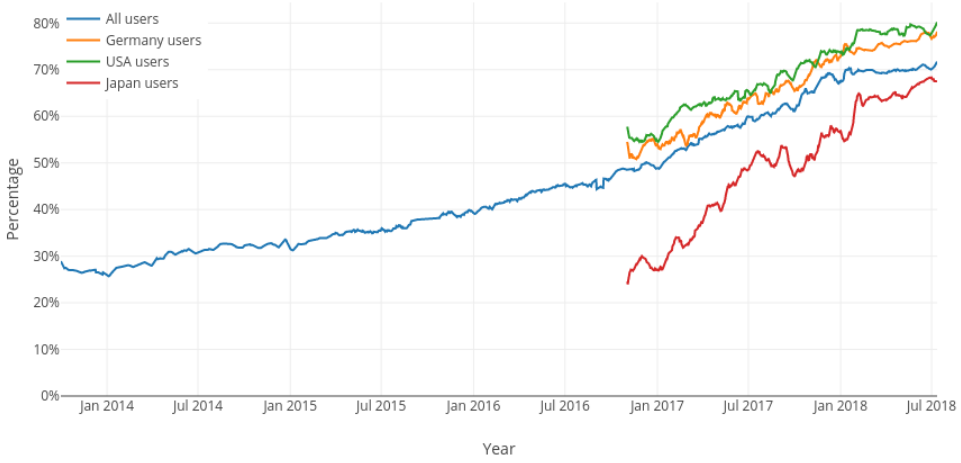

Figure 2.8: Percentage of web pages loaded by firefox using HTTPS [117], representing the increase of the *Let's Encrypt* popularity.

The old fashion HTTP was suppressed by HTTPS in web encryption searching for a more secure protocol. This movement supported by *Let's Encrypt* [6], sponsored by the most prominent companies (e.g., Cisco, Facebook and Akamai), has transformed the composition of web traffic [118]. It becomes more difficult to analyze network data, once session encryption

[6] Organization that are helping millions of sites add HTTPS to their sites for free

at layer 6 makes data unreadable, and at layer 6/7, information of protocol in use is needed for extracts relevant knowledge [110]. Thus, in the scope of this dissertation, data collected is subsequently analyzed between layer 3 and 5 of OSI model (without deep packet inspection 2.1.6), extracting meaningful material to be used in ML methods for classification and detecting traffic patterns of invasion activities.

2.5 Knowledge Extraction

As argued throughout this thesis, companies increasingly manufacture and extensive bank amounts of data. Analyzing and pull out something with meaning from that collection of data is not a trivial process. In this way, Data Mining (DM) has played an important role in the information era. DM is a process of extracting useful material from large data sets, crucial to discover underlying patterns. It can be perceived as an essential to knowledge extraction.

In a monitored network enterprise, samples collected from monitoring process (discussed in Section 3.2) will be manipulated to generate patterns that define the network action. However, the data must be prepared and treated before processing. After the cleaning process, statistical network observations are conducted to ML algorithms, that are capable of extracting relevant knowledge from the meaningless data, translating the objective of this thesis.

Moreover, in this section will be presented an overview of some knowledge extraction techniques used to audit and infer the network status. So, in an anomaly-based network intrusion detection, as described in Figure 2.9, the system is trained with samples from the monitoring traffic in order to generate a model that, afterward, is used to classify new events as atypical or not [119]. How the information is extracted from the model's fabrication is what distinguishes the anomaly detection techniques: [120]: knowledge-based (2.5.1), statistical-based (2.5.2) and ML-based (2.5.3).

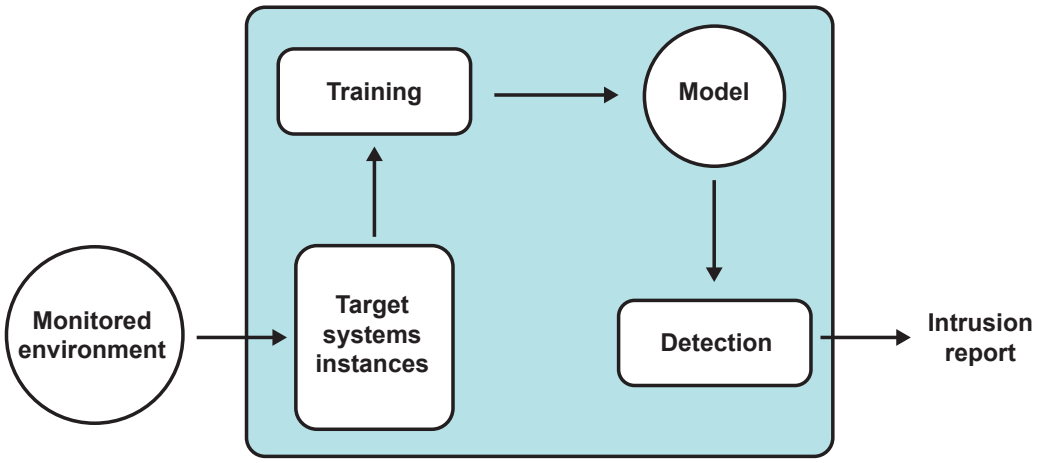

Figure 2.9: Schema of necessary steps to transform the collected data into knowledge and, posteriorly, intrusion detection.

2.5.1 Knowledge-based

Knowledge-based techniques are performed by most IDS (both *anomaly-based* and *misuse-based* 2.1.6), where network events are checked against predefined rules or patterns of attack, to easily recognize and handle with already experienced attacks. Monowar Bhuyan et al. in [14] split this technique into:

- Expert systems
- Rule-based
- Ontology-based
- Logic-based
- State-transition analysis

An example of a rule-based detection system is the open-source Snort [121], where each packet is observed according to a set of rules that can identify attacks based on IP addresses, TCP or UDP port numbers and ICMP codes [14]. However, these approaches requires a manually rule insertion, when new types of malicious traffic are observed and is crucial for anomaly detection a completeness monitor system that automatically and autonomously "learns" with new attacks signatures.

2.5.2 Statistical-based

Statistical view over the network traffic can be a simple approach showing, for instance, the most used ports and addresses, flow size or the number of packets for each protocol. This statistical metrics (e.g., mean and variance) are used to create a statistic model that represents the normal behavioral of network traffic, that is, a profile that represents its stochastic behavior [120]. Then, the monitoring data is compared with learned model, looking for unseen anomalous instances that have low probability to be generated from it. Statistical models usually are sustained on the following categories [122]:

- Operational Model or Threshold Metric
- Markov Process Model or Marker Model
- Statistical Moments or Mean and Standard Deviation Model
- Multivariate Model
- Time Series Model [123]

Despite not requiring prior knowledge of normal activities of the target system, some drawbacks should be pointed out. Once the building of a model is a prolonged process, it takes a long time to report the prime anomaly. Moreover, the selection of parameters and metrics is not a trivial process [14]. Furthermore, the network environments produce large datasets with high dimensional data. Along with this, the network is a dynamic environment composed by very differentiated data. Thus, statistical models can not address these problems [119], being necessary a self-learning system capable of adapting to a dynamic network environment and with high computationally efficacy to handle with these large sized inputs. Note that, in many cases, the applicability of ML principles coincides with statistical techniques. The former

technique is focused on building a model that improves its performance based on previous results [14], [120].

2.5.3 Machine Learning

For a better understanding of the anomaly detection approaches that apply ML techniques, it is important introduce the concept of ML and its application in irregularity disclosure process. ML is commonly associated with the futurists robots, although, it's a real and tangible concept. In fact, ML has been around for decades, nearby the 90s, with the Optical Character Recognition (ORC) and the spam filters over the world. It has now embraced several human domains such as the product recommendation and voice search. It is present on financial field (e.g, fraud detection and stock market predictions) or in the smart cities area through IoT traffic analysis and smart cars building. Also, in healthcare [124], revolutionizing, for example, radiology by improving detection for more precise diagnostics [125]. It's visible the fast escalate ability, that tends to continue, as well as, the improvement on the human life at various levels. But, will this growth only be accompanied by advantages?

According to Arthur Samuel, 1959, ML means:

> "[ML is the] field of study that gives computers the ability to learn without being explicitly programmed." [126]

A more complex and specific definition:

> "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E." [127]

However, ML many times is confused with Artificial Intelligence (AI). This two acronyms used interchangeably, in fact, have a relation. ML is a subset of AI, whereas AI is the broader concept of machines. It carry out tasks in a way that we would consider "smart", while ML, combines algorithms and methods to machines assimilated the data and learn by themselves to reach that intelligence [128].

Spam-filter is one the most usual scenario to help understand the ML concepts. This filter learns to flag spam, given examples of spam emails flagged by users and examples of regular emails, i.e, nonspam [7]. But, considers that the spam filter is built using traditional programming techniques. To complete the program the coming steps must be followed [126]:

1. Analyze and observe the problem, recognize repeatedly words or phrases (such as "4U," "credit card," "free," and "amazing") or other patterns.
2. Write a detection algorithm that covers the noticed patterns and flag emails as spam if this patterns are detected.
3. Test the program and adjust the algorithm until it produces good results.

[7] also called ham emails

It's evident that with a programming approach, as the list of rules increase and became more complex, scalability and maintenance turns into a problem. Moreover, if spammers detect that their intentions are not being concluded, they start digging deeper in the problem and discover, for instance, that emails containing "4U" are blocked. To overcome this drawback, they might start writing "For U" instead. Using traditional programming, spam filter needs to update the flag's dictionary every time that spammers write and finds new words, only being able to detect spam when spammers stop working to bypass them (thing that will never happen), entering in an interminable cycle.

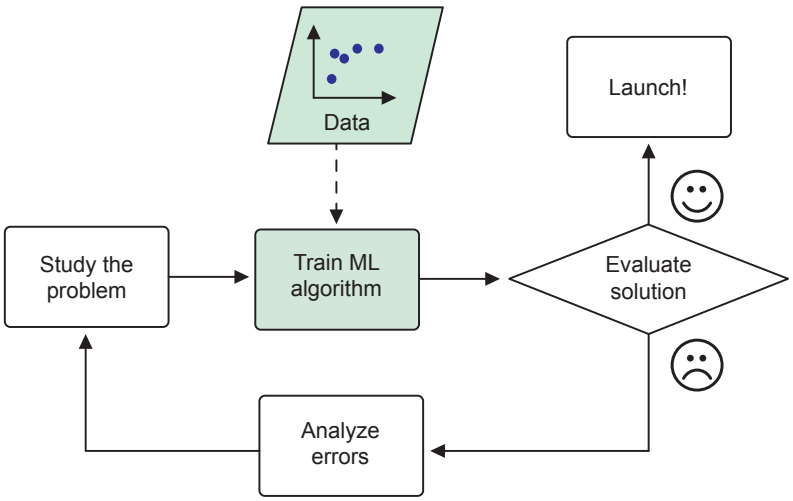

Figure 2.10: Machine-Learning semi-automatic approach [126].

In contrast, a spam filter based on ML techniques (as show on figure 2.10), using the *training set* to learn, automatically identifies which new words and sequences (that set the patterns) are a good predictors to flag email as spam or nonspam. Once this approach reaches progressive adaption, the program becomes less complex and easier to maintain, and possibly more *accurate* [126]. Also, with ML capability, the system automatically observes the emerge of uncommon words and starts to flag them without any intervention.

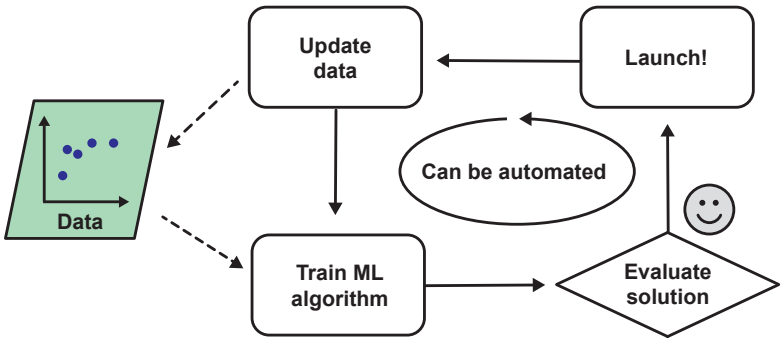

Figure 2.11: Machine-Learning schema that adapts automatically to changes [126].

The Figure 2.11, shows the automated aptitude how a training set is retained, creating almost a full automatic system, by only requiring monitoring. For problems with a reachable solution requiring long lists of rules and subject to many data changes (how is the spam filter case using a traditional approach), ML can improve the system's efficiency and control. Even for problems where there is no straightforward solution or no know algorithm, ML techniques provide tools to humans digging and highlight useful information from massive datasets, by studying what the algorithm has learned deducing patterns, which initially was not susceptible to the human eye - *data mining* (as already referenced in 2.5)

Due to the broad scope presented by ML it is convenient for identification to classify the numerous underlying system's types into categories, that will be detailed over the next sections, based on [126]:

- Influence of human supervision during training (supervised, unsupervised, semisupervised, and reinforcement Learning)
- Possibility to learn incrementally from a stream of incoming data (online vs batch learning)
- The ability to identify exclusive samples that never was seen before (instance vs model based learning)

2.5.3.1 Supervised and Unsupervised Learning

ML algorithms in the training phase may have some type of human supervision. In a training set, when each *training instance* (or *sample*) is appended to a *label* supplied by human, systems are classified as *supervised learning*. A typical supervised assignment is *classification*. Bringing the demonstrative example of spam filter again, in the training set, the algorithm learns how to classify new emails "inspecting" the class of each email sample that belongs, spam or ham [126].

Another primary application of *supervised learning* is *regression*. This task predicts a *target* numeric value, for instance, the price of a car, by giving a set of features (mileage, age, brand, and so on), called *predictors*. However, in this case, to a properly and faster learning, the *supervised learning* system needs a dataset containing the predictors and their labels (prices) [126]. Despite this, some *regression* algorithms do not have an exclusive proposal, while can be used for classification using the output value that indicates the probability of belonging to a class (e.g., 20% chance of being spam) [126].

The list of the most common *supervised learning* algorithms include [126]:

- k-Nearest Neighbors
- Linear Regression
- Logistic Regression
- Support Vector Machine
- Decision Trees and Random Forests
- Neural networks [8]

[8] Note that some neural network's architectures can be unsupervised as supervised

Distinctively, in *unsupervised learning*, the training set is unlabeled, in which the system tries to identify similarities between the inputs without any supervision [129]. *Clustering* is a familiar application of *unsupervised learning* used over different areas like engineering, bioinformatics, and medicine, where similar *samples* are gathered together, forming a cluster. For example, given a dataset with a patient's medical records and applying a *clustering* algorithm over the data, it will detect groups of similar patients. The final results could reveal subjects with related symptoms, and therefore, help the doctors diagnose. Also, *unsupervised learning* techniques can be useful for detecting data irregularities, the *outliers*, employing *anomaly detection* tasks [126]. In the previous healthcare example, this *unsupervised* algorithms could be a support for monitoring and alerting deviations on patient's condition [130].

Most problems deal with large sets of data containing a lot of complex and unlabeled data. *Dimensionality reduction*, in which the goal is to simplify the data, representing them with a smaller set of more "condensed" variables, can reduce computational load in further processing, also as, establishing a method to describe the unstructured data (i.e, *visualization*) helping to understand how it is organized and to identify unsuspected patterns [126], [131].

In the point between these two extremes is the *semisupervised learning*. In this type, the training set combines both unlabeled and labeled data. Image recognition functions presented in photo-sharing and cloud-service Google Photo's are an excellent example of this algorithms. First of all, the same person is identified over the uploaded photos grouping into clusters (unsupervised part), but then to distinguish those people, the system needs to be supervised by attributing labels for each founded person [126].

As a far cry from all these approaches is *Reinforcement Learning*, in this case, the learning system works as an *agent*, that observes the environment and according with the selected *policy*, acts. Conforming with the agent's actions, *rewards* or *penalties* are given. Thus, the system by itself learns which are the best actions (considering the given *rewards* and *penalties*) and update the *policy* until found the optimal one [126]. AlphaGo Zero, the Google's DeepMind supercomputer, beat the world champion at the game *Go*. It was programmed with millions of moves of past masters using *reinforcement learning* algorithms, in which the winning policy was learned and tested by playing many games against itself, and used in the ultimate final [126], [132].

2.5.3.2 Batch and Online Learning

Another criteria used to classify ML systems is the capability to or not learn incrementally from a stream of incoming data. *Batch learning* is incapable of learning incrementally. In the training phase the system uses all data at once, and when new inputs arrive, the *training set* needs to be updated with the fresh data. Due to the system's limitations into adapting, a new system version must be done from scratch every new instance, being time and computing consuming because it is done *offline*.

To solve this impracticable solution in production environments, where a considerable need of computer and storage resources are needed, *online learning* comes up, slowing down the learning process and enabling the learning incrementally capacity. In this case, the system

at a training stage, instead of collecting all data at once, the *training set* allows the provision of new instances, individually or by small bunches, called *mini-batches*. Comparing with *batch learning*, it supports and easily adapts to fast-paced environments where data is continually changing, improving the time learning and not being necessary to rebuild the system for each modification. Furthermore, the segmentation of the incoming data can be discarded once the system has learned new data, reducing drastically the computing resources used.

However, it is mandatory to take into account the *learning rate*. This parameter controls the system's adaptability to data change, calibrating its reaction facing the late data. When the system has a high learning rate it quickly adapts to change, but also quickly forgets old learned data. That is what happens in a spam filter when built with high learning rate, identifying only the new spam models. On the other hand, setting a low learning rate, the system will learn more slowly, at the same time that will present inertia, i.e., will be more tolerant to noise in data, not detecting possible errors. It is notorious the influence of corrupted data on the system's performance. Imagine a malfunctioning sensor on a robot. The faulty data may sabotage the normal output results, translating into the loss of time and resources (both monetary and computational). So, *online learning* monitors solutions by tracking every change and scans the training set before being applied to algorithms. This strategy overcomes the anomaly data storage and the system's accuracy and efficacy. [126]

2.5.3.3 Instance-Based Versus Model-Based Learning

The already described types of ML tasks have one common goal: making predictions for new values. Although, the system performance depends on how it generalizes, that is, the ability to identify exclusive examples that never was seen before. Thus, there are two generalization forms: *instance-based learning* and *model-based learning*. In the *instance-based learning*, the system learns trivially, learns by heart [126]. For instance, the spam filter could be built based on a simple approach, flagging only the emails that have already been flagged. However, as this is not fully effective, *instance-based learning* uses a *measure of similarity* (e.g., word count, etc.) to flag emails that are analogous to known spam. In opposition, *model-based learning* handles the examples of the *training set* to generate a model which is used later to make predictions by regression. Moreover, to the judge model's quality, there are performance measures like the fitness and cost functions. They are effective to evaluate how close the predicted solution by the model is to the known values ("ground truth") [126].

2.5.3.4 Main Challenges of Machine Learning

To benefit from the ML potential, the first thing to focus on is data. A good dataset might convey in a good prediction model. But not always. The algorithm choice is also a critical step to a good model. However, when they are put in practice into the wrong way, may put at risk the problem resolution.

Insufficient Quality of Training Data Michele Banko and Eric Brill in the acclaimed paper [133] inspect the effects of data size on ML for natural language disambiguation problem [9]. The final results demonstrated that different ML algorithms, against a complex problem, performed almost identically, well once they were given enough data [126]. But now comes up the following question: how much is the "enough" data? There is no generic rule. The corpus size depends on the problem's complexity. However, the amount of data is critical for good learning, since even for a straightforward problem typically thousands of examples are needed, and as stated in [133], the effort direction should be to change the development of the corpus. The same idea was defended by Peter Norvig et al. in [134] showing the importance of having a good dataset facing the algorithm development.

Non-representative Training Data Big Data trend [135], implies huge datasets. Typically goes side by side with ML, but to ML successfully accomplish its duty, that is, to predict various outcomes for new data, it must generalize appropriately. Having a vast corpus does not mean that the data is representative of new cases to generalize. This trade-off usually is hard to happen, as demonstrated by the famous example of sampling bias (i.e., the samples can be non-representative if the sampling method is distorted) during the US presidential election in 1936, that has opposed Landon against Roosevelt. In a short version, the Literary Digest conducted an extensive poll, sending mail to about ten million people, and predicted that Landon would beat Roosevelt 57% to 43%. As it turned out, Roosevelt won 62% to 37% [136]. The problem was on the sampling method. Despite a large amount of collected votes, the poll of Literary Digest magazine surveyed its readers, and the readership was skewed in a subgroup that supported Landon, in which a non-representativeness model was created incapable to making a proper prediction [126].

Poor-Quality Data Nevertheless, an extensive volume of training data is not synonymous of quality. If the *training set* is incorporated with flawed and imprecise examples bringing up noise and outliers occurrences, the most likely outcome is the system's inability to detect underlying patterns, having a weak performance. The training corpus demands a in-depth examination, discarding the evident outliers, or when there are instances with missing features (e.g., lack of age or weight values), they can be discarded, ignored or filled with other values (e.g., with the median age) to create a clean and solid *training set* [126].

Irrelevant Features A ML system only is capable of learning if the training data contains more relevant than irrelevant features. As Aurélien Géron wrote in [126], "garbage in, garbage out", emphasizing the importance of having a good set of features for ML. This process called *feature engineering* is expressed on the successive points [126]:

- *Feature selection*: Election of the most significant features.
- *Feature extraction*: Create relevant features using the selected ones.
- Discover new features by, continuously, gathering and processing new data.

[9] For example, distinguish whether to write "to", "two" or "two"

Overfitting and Underfitting the training data A common problem in ML is when a model performs well on the training data but generalizes poorly to any new data, most of times related to complex problems. This is known as *overfitting*. The opposite, is *underfiting*, that occurs when the model is very simple and performing incorrectly on the training data [137]. The selection of additional parameters and other suitable features (*feature engineering*) or reduction of the regularization of data [126], are the best suitable tools to overcome these drawbacks.

2.5.3.5 Testing and Validating

To know how well the ML model generalizes we have to test with new cases and not merely with the *training set*. Hence, to support this concept, normally data is split into two sets: the *training set* and *test set*. Training the model with the *training set* and testing with the *test set* helps to evaluate the model generalization behavior using the *generalization error* obtained by evaluating the model on the test set when confronted with new cases. So, *generalization error* reveals how the model performs on instances that it has never seen before, although, validating different models to the same testing set falls into overfitting. That is, always dealing with the same data can adapt the model to that training set, taking them to not perform so well on new data.

To avoid this problem, the training set is split into smaller sets, where each model is trained with a different set of combination and tested against the reaming ones - the *validation test*. his technique that allows reusing of the training set is known as *cross-validation*. So, after *cross-validation* is completed, the final model is trained and ready to compute the generalized error measured on the test set [126].

Additionally, *No Free Lunch theorem* [138] is a typical example used for the validation case. Assuming that each model is a simplified version of observations, i.e., when does not exist any assumption of the data, there's no reason to prefer a model over any other. For instance, it may be a linear regression or a neural network, that there is no guarantee à priori that any model is preferred over another, being the only way to know which one performs better and if which one works for each case is to test them all [126].

2.6 Existing tools and solutions

Along of this work has been described some security mechanisms implemented by corporations at the network level, being powerless to cover and stop all attack attempts. So, network anomaly detection techniques are critical to safeguarding the internal peers, as well the classified and private data for being disclosure. Among the several existed methods to detect atypical network behaviors, for example, the interpretation of security logs and honeypots, this state of the art will focus on low-level network traffic analysis mechanisms. Moreover, the described network traffic anomalies solutions will be divide into non-commercial (corresponding to implementations carried by conferences papers and articles) and commercial systems, directing

the research scope to the two anomalies actions endorsed by this dissertation, propagation attacks, and data exfiltration invasions.

2.6.1 Non-commercial

2.6.1.1 Spreading threats

Dainotti et al. in [139], study the network behavior of worms analyzing statistical properties of traffic generated by these spreading irruptions. More precisely, was studied the Witty and Slammer worms campaigns, computing packet-level variables, such inter-packet times and packet sizes, from each session and over all aggregated traffic. In this experience the capture traffic was filtered, dealing only with UDP packets, with source port 4000 for the Witty case, and port 1413 for the Slammer worm case. Also, DNS traffic (related to legitimate data) was permeated allowing to compare the mismatched proprieties between them. Important to note that all the traffic traces contain only packet headers until layer 4, that is, no payload information is retrieved. To analyze and extract the information from packets was used the open-source software Plab [140][10], capable to separate into traffic sessions. So, for each session the inter-packet time and packet size was calculated. Moreover, same information was computed at the traffic as a whole. After collected this metrics, statistical methods were applied over them, to visualize and characterize worm influence:

- Marginal distributions of inter-packet time and packet size for each host-based session was studied using the Probability Density Function (PDF) and Cumulative Distribution Function (CDF) [141].
- Packet size and inter-packet time autocorrelation of a infected host and DNS server.
- Wavelet analysis [142] over the aggregate traffic.

The applied statistical measures in this work were succeeded regarding behavior and action mode detection of a worm, revealing peculiar distributions, lack of temporal correlation comparing with normal traffic. Thus, this study is a useful approach to give insights into the worm's impact on network traffic. However, a posterior classification process is needed to generalize for unseeing spreading campaigns.

In the same way, Wagner et al. in [143] aimed to detect spreading behaviors by network traffic inspecting. It was developed an entropy based analysis method that highlights outbreaks of fast worms in near real-time, that from the author's perspective, can be empowered to catch other network anomaly actions. So, from the observed network, packet information on flow-level was extracted, that is, the IP source/destination address and the source/destination ports belonging to a flow (TCP connection or UDP data stream). Wagner et al. [143], used the measure of entropy to disclose regularities in connection-based traffic. Entropy [144] is a basic concept of information theory which measures the uncertainty of a collection of data items: more random it is, the more entropy it contains. In this way, entropy can be related to worm behavioral because when a fast scanning worm propagates, the infected host tries to find

[10]Software belonging to the authors of this work

other hosts to contaminate adopting a random style. Entropy is also related to lossless data compression: the theoretical limit of the compression rate of a bit sequence is precisely the entropy of the sequence [145]. Using this assumption, the sequences of network measurements are compressed, where the resulting compressed size serves as the entropy estimation. In this experience, Netflow v5 was employed to collect network data and extract measurements, being realized two real-world proof-of-concept examples using the Blaster and Witty worms. In addition, a sliding window strategy [11] was used to reduce the system latency and achieved real-time analyzes. Therefore, results reveal, in the case of a scanning host, the overall entropy in a specific time window is subdued to a change. More precisely, the presence of many flows with the same source IPs (the scanning host) will lead to an abrupt decrease of the entropy in the distribution of the source IP addresses. At the same time, the scanning host will attempt to contact many different destination IPs on (possibly) different ports, generating an increase in these entropy measurements.

In [146], Taha et al. also based on entropy metric to build a real-time system that analyzes network traffic and detects anomalies. Similarly to the experience performed in [143], from traffic flows were extracted the pairs source/destinations IP address and ports number, however, the authors, also considered the number of bytes sent and receive as a feature. So, packets were captured during a period of time, where the entropy was calculated in each time interval. In this experience, in order to differentiate between normal and abnormal network traffic behavior was introduced an adaptive threshold, defined by the degree distribution and mean of the entropy values. In this way, when the entropy value of network traffic flow exceeds the threshold, then it constitutes an anomaly. Besides, Yao et al. [147] employed a combination of information theoretic entropy measure and Random Forest classification to detect anomalies in network traffic. They used only four features from the packet header (source IP, destination IP, source port and destination port) collected during a specific time window. This exercise concerning the research works performed in [143], [146], can be seen as a solution more adaptable to new types of anomalies since it empowers ML techniques.

Entering on ML field, new approaches are rising to make the anomaly traffic detention more accurate. Sarnsuwan et al. [148] proposed an implementation to detect propagation incursions by processing the network packets data and extracting features from abnormal/normal traffic data, using three different supervised ML algorithms for data classification: Bayesian Network, Decision Tree and Random Forest. In this method, Blaster worms are considered and inserted into a LAN along with DoS attacks and other port scan attacks. Contrarily, the normal data were obtained from web browsing, instant message, etc. Some features were extracted from raw data such as the IP address, port, protocol and some flags of the packet header. Sarnsuwan et al. [148] also applied a feature selection method based on the information gain technique to select the more representative features. This framework works successfully, but cannot to produce a perfect worm detection as some amount of actual data is not classified [149].

Limthong et al. [150] analyzed which interval-based features of network traffic were feasible enough to detect anomalies and the more accurate ML techniques for the disclosure process.

[11] The same approach was adopted in this dissertation as will be described in 3.2

The authors select several features (e.g., number of packets, sum of packet size, number of flows, number of source address, and so on) computed during a defined interval, and found which of them were more appropriate to detect back, sumrf, ipsweep, neptune and portsweep attacks, through the use naive Bayes and k- nearest neighbor algorithms for the traffic classification. Although, this work cannot perform real-time anomaly since it only studies the best features for each outbreak. Moreover, Limthong one year later in [151], continued the previous implementation, proposing a novel framework based on a combination of time series and feature spaces, able to perform real-time detection over a real network environment.

2.6.1.2 Data leakage

Data leakage, or data exfiltration, is particularly hard to detect because it uses a variety of techniques to make the exposure more imperceptible. Even so, it is essential to research new approaches to overcome and attenuate the damage caused by these outbound incursions.

Al-Bataineh et al. [152] proposed a novel technique to analyze and detect the malicious data exfiltration in web traffic. It inspected the spiteful data stealing behaviors of one of the most well-known data stealing botnets, Zeus. By studying its network activity, the authors proposed a classifier to identify HTTP request that transports stolen data. So, the goal is to monitor HTTP requests and identify malicious demands, employing the entropy and byte frequency distribution of content files as features. In this work, encryption is seen as an opportunity for detection rather than a limitation because if encryption is detected where it is not expected, it should be considered an anomaly. In this way, Shanon's Entropy [144] was used to check the presence of encrypted contents in POST requests through the byte entropy computation, once encrypted contents looks random in the ideal case, established a threshold-based classifier: if the content entropy were equal or greater than the threshold, then the content would be classified as encrypted, otherwise its considered non-encrypted. To reduce false positives and negatives carried by the byte entropy feature, the byte frequency distribution was introduced as a feature. It is a concept already proposed not only in file type detection problems but also in malware detection inside packet payloads [153]. However, in [152] the ambition was to differentiate encrypted contents from all other types given that it has the most uniform distribution. To handle with this feature, from network web request and responses, TCP flows and application layer data were assembled, parsing HTTP traffic to extract headers and contents of each request and response. So, entropy and byte frequency distribution were the final features, that fed the three different algorithms: NaiveBayes, Multi Layer Perceptron (MLP) and J48 decision tree algorithms [12]; being the last one the most accurate, revealing good results on real scenarios.

Furthermore, Liu et al. [155] developed SIDD [13], a three-stage framework for detecting insider attacks and sensitive data exfiltration. This system corresponds to a network bridge

[12] Weka software [154] was used to apply the three algorithms

[13] Sensitive Information Dissemination Detection

located at the edge of the protected network which scans the outbound traffic, being composed by three main levels:

- Application identification

  This is the first stage, where the system determines which applications, based on characteristics of its flow (at network level), are being used. To classify the applications was used frequency domain and wavelets [156] analysis over the network traffic attributes, such as the packet inter-arrival time and packet size. Application identification can provide some useful information which allows proper policing of application traffic to dictate what traffic is allowed across the network boundary.
- Signature and content detection

  After completed the previous phase , DPI techniques were empowered to analyze the network data content. The idea was to create network traffic signatures from its content allowing to compare signatures and detect data exfiltration, based on packet volume feature. Afterward, the retrieved content information is treated as a time series which allows to use signal processing techniques and tools to analyze these signals.
- Covert communication detection

  The last level was responsible to detect covert channels, where attackers mask the protected information using legitimate channels, for instance, steganography scenario.

Although SIDD [155] applied statistical and signal processing techniques on traffic flows, the major drawback of this system is the high packet processing capacity required in each level, requesting a tremendous processing power to monitor and analyze each outgoing packet at runtime. Also, the system dependency on the content signature matching, making it susceptible to be outbreak by dissimilar types of data.

Ramachandran et al. [157] developed an implementation for data exfiltration detection over the network by building a behavior based model. This experience does not rely solely on network traffic analyzes but also the overall system features like the total memory and total CPU consumption level. Since these last features do not part of the low-level traffic analysis focus, the study of the work performed in [157] will only consider the network aspects. Following these lines were employed the number of outgoing packets, outgoing bytes, incoming packets, incoming bytes, control and data packets as network features, collected using SNMP. So, initially, the system learned the normal behavior of a network by using kernel density estimation (KDE) methods [158] over the described data network metrics. Afterward, in the testing phase (that is, in the detection chapter) once again the kernel density estimation is computed for the real-time traffic. Further, the current KDE values are compared with the learned ones by check how much they are correlated: when KDE values were weakly correlated, it is probable that anomalous actions happened.

Similarly, Sharma et al. [159] described a framework to detect potential exfiltration events by monitoring a set of system and network level features (i.e., from hardware to the application layer). Several distributed sensors composed it, each one responsible for a specific monitoring task (e.g., hardware, process memory, system calls, network, etc) that continuously compares live data with the normal-behavior profile. Once again the work analysis will focus at network

level. To model the outbound network flow characteristics, the mean-inter departure packet times (the rate of packets flying out) and the number of packets in a single TCP session (quantity of outbound data) were chosen by the authors as network features. So, after the egress traffic has been sniffed, over the aggregated data was computed average value of the inter packet departure time per TCP session and the average number of packets sent in a single TCP session, along with a whitelist carrying all the IP addresses with which the host communicated. Moreover, the framework performance was compared against six commercial security software, that in general, have better results for detection of the attack vectors simulated in this experience. Although this implementation presented high discover capability against the commercial solutions, some extra work, such as more exfiltration dynamics, where leakage attacks deal with different size of files, may increase the framework performance, since among the 1154 monitored TCP sessions, only three sessions involved the attacker exfiltrating small files (<1 Mb) from the victim's machine.

2.6.2 Commercial

Open source IDS are increasingly being used due to the free availability and simple usage. The most widely utilized are: Snort [160], Bro [161] and Suricata [162]. All of them were designed for intrusion detection, yet, Bro mostly could be considerate as a traffic analysis language [110]. Their detection engine, where is detected any intrusion activity exiting in packets, is signature-based (2.1.6). This means that each packet is opposed and compared against a set of predefined rules, that is, hand-crafted signatures of recognized attacks. The major drawbacks of this open source solutions are:

- **Inaccurate detection**

  First of all with a signature-based classification these monitoring devices cannot detect zero-days exploits (2.1.6). Moreover, it is necessary a system with high exposure and low false-alarm rates.
- **Poor scalability and time delay action**

  Especially when there is increasingly more traffic, the exponential growth of data is causing tool overload. Taking each packet and verifying if some of the rules matches, could degrade the performance of the system along with the increase of resources to perform the task.

It is noticeable that most of today's IDS failed. As follows, recently Cisco launched a new software, named *Cisco Stealthwatch Enterprise*. It provides continuous real-time monitoring, capable to detect suspicious behaviors such as:

- zero-day malware
- DDoS attacks
- lateral movement outbreaks
- insider threats
- data exfiltration charges

This implementation uses cognitive analytics, i.e., a cloud base analysis engine that assembles a baseline profile to automatically detecting anomalous traffic. Cognitive analytics is an exclusive factor since the processing core is moved to the cloud, where ML and statistical modeling techniques are applied for intraflow metadata and encrypted traffic analysis. In this way, the system can learn without human support, from what it sees and adapted to changes [163].

Darktrace's company developed a competitor software, the *Enterprise Immune System*, that combines real-time threat detection employing ML algorithms. It determines the normal of each user and subnet in the network, and with a self-learning process, it can detect traffic deviations. Lately, was released the *Darktrace Antigena* for a faster and precise response when facing emerging threats [164]. It succeeded spited a real threat executed by a resentful employee that reacted to their company strategy to deal with Brexit, by digging out and leaking confidential documents. So, Darktrace's Antigena detected and stopped this insider data exfiltration action [165], expressing its potential and the anomaly detection aptitude in true corporate network scenarios. In fact, this two software endorse a ML learning system capable of detected the to main goals oh this work, malicious propagation inside a network and the data exfiltration scene.

2.7 Conclusion

To conclude, the examination conducted in this chapter revealed that in the face of existing flaws in modern corporate networks, not being able to fully ensure the integrity and confidentiality of data and services, the usage of ML associated to anomaly detection to complement those points of failure, meets the current trends presented in Section 2.6. However, the high variability and unpredictability of network traffic can influence the decision of the ML system due to the high variance introduced. Along these lines, being the path to follow in the anomalous domain, human intervention and supervision is a less required factor in autonomous decision, but for now not yet totally independent.

CHAPTER 3

Methodology for detecting anomalous communications on a corporate network

In this chapter is discussed the possible methodologies used to infer network patterns and expose anomalous behaviors inside a corporate network. It is a generic procedure, which covers not just one particular case, acting as a guide, where the collected raw network data is transformed into relevant information capable of distinguish between normal and abnormal observations. The first section explains how data is gathered, following by the description of the used knowledge extraction method. This chapter is also a pipeline to recognize network patterns, from the data visualization and interpretation, to the ML algorithms application.

3.1 Network data acquisition

Monitoring probes location is the first concern for corporations being able to infer network behaviors. As discussed in Section 2.4, among the several monitoring tools and strategies, aiming to detect anomalies that exceed the normal traffic patterns (e.g., lateral movement and data exfiltration incursions), at an enterprise level, a distributed port mirror solution applied in all access layer switches provides a detailed and precise acquiring traffic data. If it was placed at distribution and/or core layer switch level, network data generated by attached end-hosts on the access layer will not show up. On the other hand, gathering network data at host-level, from each existing machine and equipment, sounds like an impractical strategy facing the increased complexity in the analysis and storage of the collected traffic.

Thus, operating at layer 2 level of OSI model, it is possible to mirror internal VLAN traffic that passes through an interface, performed by the associated endpoints, pointing an unusual contact between direct machines, which may be indicative of internally spreading malware. Besides, handling with granular information (low-level traffic) allows learning

the organization's normal outbound communications preventing information out-of-company information migration. However, to clarify, the place and type of monitoring strategy varies according to the problem in question (as discussed in Section 2.4).

Sniffers can perform traffic analysis functions, withal, it is the intent of this dissertation, that probes only capture data without any processing ability. In this way, at a sniffer-level, is generated a considerable volume of data containing nonessential information that can be discarded, simplifying the analysis and storage process. To build a behavior analysis-based systems it is necessary to dissect packets in order to extract symbolic data that express network activity. The confines of packet inspection are defined by the fourth OSI level, reading and collecting mostly values of OSI layer 3 and 4 (Network and Transport respectively), relying on TCP and UDP protocols for obtaining the fowling packet information:

- Size of the packet
- IP origin and destination
- UNIX epoch time of the packet
- Ports origin and destination
- TCP flags (only for the TCP packets)

3.2 Feature engineering

3.2.1 Converting packets into samples

The wired data collected from the probe needs to be analyzed, since, only represents network packet metadata. It must be translated into network traffic observations to create a model that describes a pattern or behavior. Some models could be defined by few rules that can be done pragmatically by looking into the dataset and delineate activity thresholds. Still, when dealing with extensive and complex datasets the analysis becomes much arduous, being ML ideal to discover hidden patterns and unrecognized relations. The ML model performance depends on the representative examples provided to it. That means, to the model predict more correctly the normal and the anomalous patterns, needs to be fed with examples that illustrate activity observations. In this way, determining what ML algorithms input are is known as *feature engineering*. This is the process of transforming raw data into features that represent the underlying problem, many times dependent on the analyst's experience and of the dataset understanding.

Shaping the network relations and behaviors from the extracted packets information, by creating the features could be done using several approaches. However, packet data can be associated over time, revealing periodic actions and quantify activity events. Both human and machine acts occur at a different frequency, rate, and scale, but, may present recurrent actions over time, translating into periods of similar activity. For instance, a human user performs more actions during the day than at night, characterized by small and sudden traffic peaks corresponding to the clicks. On the other hand, a company file server may present a more disperse traffic rate of activity during the day, producing higher action scales with more

frequency than human behavior. Nevertheless, network events do not strictly exhibit periodic behavior and can vary over time. So, this variation also should be taking into consideration to create more reliable features.

In light of these time-dependent examples, packets could be modeled in relation to time tracing an activity pattern and revealing underlying services. The main idea is illustrated in Figure 3.1, where the total time of activity (i.e., the capture time between the first and last packet collected by the sniffer), is divided into Δt small intervals.

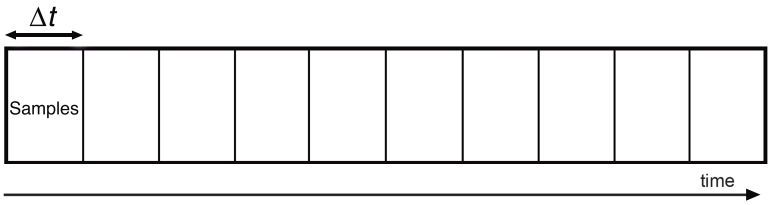

Figure 3.1: Splitting of the total activity in smaller sampling windows. Each slice aggregates the sum of packets metadata during the Δt time.

Each slice contains counting metrics of data over Δt period, that will be used to form the features which traduce network observations. So, the capture time is divided into *sampling windows* (corresponding to a Δt slice), each one encompassing the sum of the packet events occurred during that interval. These metrics will model an activity pattern using some "individual" time-dependent attributes such as:

- Statistical packets counts

  Counting of packets along with their attributes like the number of contacted IP addresses can define a normal or intrusion behavior.
- Network protocols routine

  Packet sum by protocol and their utilization, for instance, number of UDP/TCP ports, could reveal a pattern of services used and its position in the network

However, packets can be aggregated into traffic objects that have the same IP source address and one of the TCP port numbers equal, the data stream aggregations [166]. Empowering this aggregation solution enables the control of all ingress and egress traffic, managing its volume, duration and the open sessions with other peers. Thus, counting metrics can be computed over these flows. Since an exposed port characterizes a service„ that means, uses TCP or UDP at the transport layer, it is possible to manage the number of connections and service usage by machine. Yet, the way of how TCP [1]traffic is aggregated depends on the situation. In the case of supervising the internal network state, a good approach is counting the overall TCP sessions, more precisely, the opened sessions for all contacted destination ports. This is useful to detect propagation attacks inside a corporate network. Differently, if the goal is to control a particular service, then seems a good solution to count the active sessions for the specific service port (for example, port 445 related to SMB service). Further, concerning outbound traffic, it is necessary to take into account the firewall permissions. At

[1]Most of the Internet applications runs over TCP

the corporate level, some external services are "blocked" in the firewall, so, in this case, only the permitted ports should be included for the TCP sessions count.

Other counting attributes can be used. However, the choice depends on the environment type, the problem and what is intended to be observed for modeling its behavior. For instance, if the environment in question deals with heavy and abundant data, like the cinematographic companies, where terabytes of data travel over the network, counting the amount of traffic per interval could not give significant observations. Instead, identifying traffic flows and services usage could be a good starting point to find patterns. On the other hand, enterprise environments contain merely traffic associated with mail and browsing services, considering the volume of traffic may be a way to detect unusual behaviors.

Other significant concern is the value of Δt. Assuming short values in the splitting criteria, less data is accumulated, and consequently, it is ideal for detecting high-frequency events, such peers communications. Nevertheless, high Δt values create small sampling windows with more data, capable of detecting human events.

3.2.2 Generating network observations

The extraction of samples from packets is the first layer of computation, where the result is merely the sum of counting metrics. In this way, it is necessary a next computation-level, where these statistical metrics are transformed into significantly network observations, capable of modeling an activity behavior and pointing unnoticed deviations. This is the last step before the ML data pipeline is applied, and for this reason, the modeling process efficacy is associated with the ML system performance: generating meaningful network observations will facilitate the ML algorithm to produce representative results.

Therefore, in this stage, the sample windows created in the previous process are combined through a sequential iterative process. At each iteration results in a new *observation window*, being a high-level computation of the previous sampling windows. Assuming that the data was captured during one week and each sampling window contains statistical data during two minutes, Table 3.1 shows the result of a weekly strategy of data splitting.

Table 3.1: Relation between observation window's length and number of observations. It was considered one week activity with two minutes sampling windows.

Observation's duration	Formed windows	Observations number
10 minutes	5	1008
15 minutes	7	672
30 minutes	15	336
1 hour	30	168
10 hours	300	16
24 hours	720	7

The way of iteratively is divided will influence the system response time and the number of network observations produced. Assuming that no splits were made over the capture time equal to one day, thus, Δt will represent a day and only will be produced one observation.

Moreover, the system only will be capable of comparing the observation, identified if it is a normal or abnormal pattern, after a day ending, meaning that the response time is equal to one day. On the other hand, if the total day of activity collected is highly split, the response time will increase. Withal, observation windows will contain more non-representative data, since the distinctive patterns will be dissolved among the several observations that provide very similar data. Along with the response time concern, the number of observations generated is also an important point, since some ML algorithms need more or fewer data according to the problem. In this way, should be found a balance between the system's response time and the number of observations needed.

Furthermore, the described sequential iteration strategy, where windows are inflexible blocks, cannot handle activity variations over time. For example, a company file server configured to every day, at the same time, perform backups of a workday, and for some reason, a system error occurs changing the period of backup. So, this activity is seen as an anomaly when shifted to other windows. With all these issues, a sliding window strategy, as described in Figure 3.2, it is preferable carrying the following benefits:

- Increase of the observations number
- Increase of the response time:

  The period of decision is defined by the shifter value, which traduces the dislocation time between windows. It can cover longer periods of observation maintaining a short period of decision, allowing quicker responses.
- Equal portion of sampling windows covered:

  By doing several shits over the same data, distinct observation windows are formed mapping the same behavior, and the activity variations are no longer consider abnormal action.

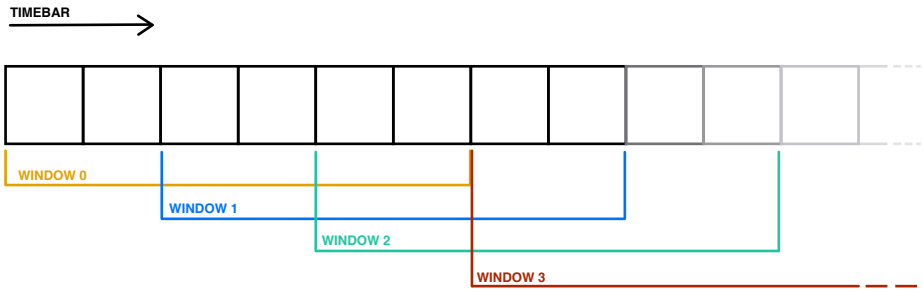

Figure 3.2: Representation of the sliding window implementation over the various sampling windows. In each iteration a new sliding window overlaps a set of windows, generating network observations.

Thus, as shown in Table 3.2 and comparing with the results obtained in Table 3.1, when using a sliding window strategy an increase of the observations number is verified. However, as in the previous procedure, the trade-off between the observation window duration and the shift value should be taken into consideration, to find the right balance between them.

Table 3.2: Number of observations produced variating the observation window duration and the shift value, when a sliding window strategy is empowered.

Observation's duration	Shift time	Observations number
10 minutes	1 second	604201
15 minutes	10 seconds	60391
30 minutes	40 seconds	15076
1 hour	10 minutes	1003
10 hours	1 hour	159
24 hours	4 hours	37

Correspondingly, the system's scope has a crucial impact on the sliding window's decision size. For instance, a smaller sliding window would not be able to fit HTTP browsing traffic since it is characterized by short periods of high bandwidth utilization with non-periodic durations. On the other hand, hourly periodic backups would be detected using a larger sliding window.

3.2.3 Network modeling and feature conception

After the construction of the sliding windows, it is needed to convert the statistical samples into features. In this way, inside of each sliding window feature computation is performed. As shown in Figure 3.3, each column represents the individual counting attributes generated during the sliding window interval. One strategy to extract the pretended network observations is to apply time-independent statistics like the *average*, *median*, *variance*, etc; over each column. Supplementary, another view for modeling the network activity is to count the number of zeros for each attribute column. For instance, considering the packet's count attribute, the presence of zeros reveals the lack of activity in that period and, afterward, it is possible to generate extra features like the silence period, along with their mean, average and so on. Instead, the counting of zeros over all the sliding windows (i.e., from all columns) could exhibit the window's completeness and shape.

In this way, the next step is to provide these features to the ML pipeline in order to build a profile based on pattern identification, used to classify and find unusual activities.

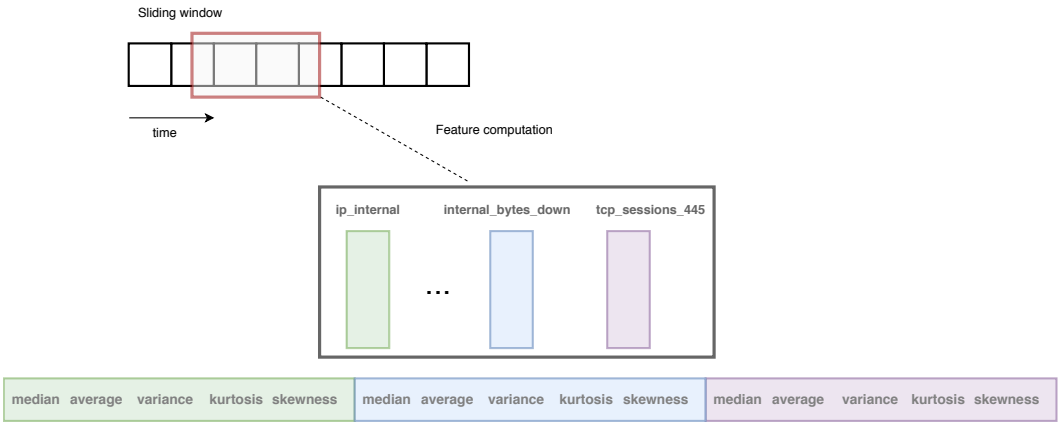

Figure 3.3: Illustration of how feature computation happens inside of a sliding window and produce the final observation features.

3.2.4 Dataset interpretation

After the feature creation process is completed and representative activity observations are established, the formed samples represent statistical values that will feed the ML algorithm. Since it is a supervised learning problem, the dataset needs to be labeled for the ML system to classify anomalous behaviors.

A high-level overview of the data is a good starting point to interpret the structure and the composition of a dataset. Checking the number of instances per feature or label, along with the mean, minimum, maximum, standard deviation and percentiles gives a view of the dataset shape and completion. Standard deviation and percentiles measures show how to disperse the values are from each other, and, in its turn, counting the number of instances that can reveal if the dataset is or not balanced. Another way to have a perception of the data is to plot a histogram that shows the number of instances (on the vertical axis) that have a given value range (on the horizontal axis). When performed over a period of time, can point out the silence and activity period, along with the samples scales and the data distribution. This last point is important since when histograms extend much further to one of the sides in relation to the median (tail heavy) can be more difficult to ML detect patterns. Figure 3.4 represents an example of a histogram showing the number of collected network packets when Youtube's videos were seen. It is notorious the activity burst between the 120 and 150 seconds, unlike the inactivity that marks the remaining time.

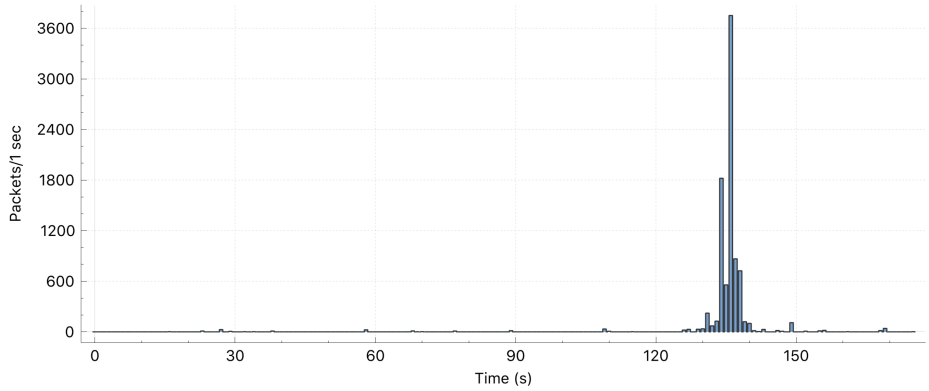

Figure 3.4: Histogram obtained from counting packets when Youtube activity was realized.

Considering that features are collected and generated with the network model in mind, there is no need to clean the data. A typical step is to fill the missing data and remove the malformed ones (e.g., null values and presence of characters). In case of classifying network anomalies, the samples resulted from the sliding windows computation, exhibit instances with the value zero, being essential to model a pattern, once traduced the lack of activity during that interval.

Before loading the data to the ML algorithm, it is also relevant to look for correlations between attributes. Computing the standard correlation coefficient between every pair of features can expose uncovered relations among data. Figure 3.5 exemplifies the correlation matrix of three features. The main diagonal (from top left to bottom right) would be full of straight lines (each feature directly correlates with itself). Thus it is plotted the representation of a histogram of each diagonal feature [126].

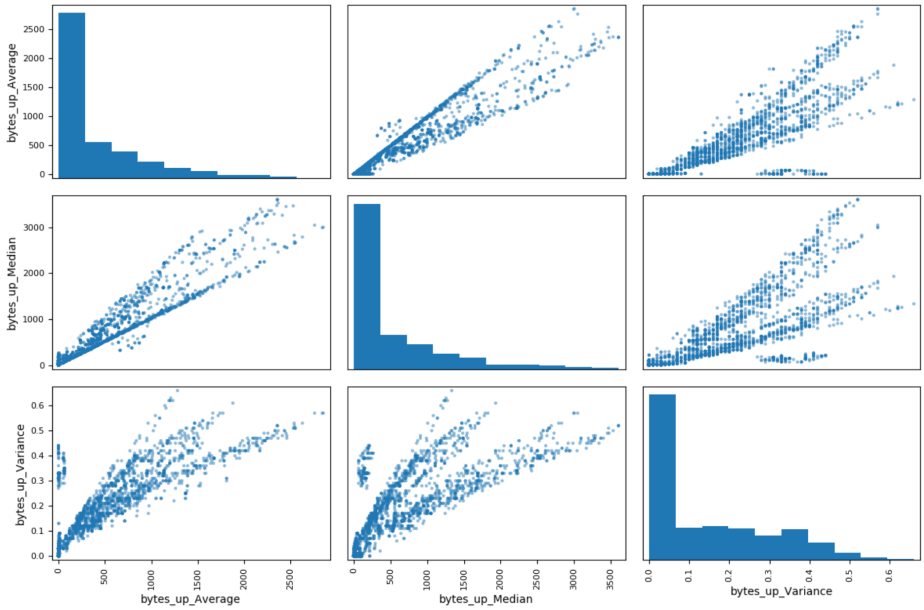

Figure 3.5: Correlation matrix of some dataset features. Diagonals symbolize the feature histograms where it is visible the tail heavy problem. Graphs show the linear relationship each pair of features aiming to find correlations for each pair.

From the linear correlation matrix shown in Figure 3.5, it is noticed that bytes upload average and bytes upload median features are correlated due to the density shape caused by the overlapped points. Besides, looking to the diagonal line of each feature, it is verified the tail heavy distribution problem already mentioned. In this way, linear correlations allow combining high correlated features, avoiding its redundancy.

However, searching for data correlations can be a complex task when datasets are too large, being difficult to see all the paired relations. For instance, if the dataset has 15 features the scatter matrix will form 15^2 plots, making it visualization impossible.

3.2.5 Features dimensionality

Having an extensive quantity of features does not make an estimator more effective. In fact, the performance of a classifier decreases when the problems dimensionality increase (i.e., the number of inputs rise), along with computational performance on the training phase. No fixed directive tells how many features should be used in classification, being dependent on the amount of training set available and the classifier type used. In this way, for a good estimator performance, fewer features require a fewer number of samples, whereas, the number of training instances needed grows exponentially with the number of dimensions used. This is the *Curse of dimensionality* problem. In light of this, an infinite number of samples, in practice, is impossible to obtain, once is important to understand the required quantity of features concerning the amount of existing samples, always having in mind the classifier performance.

Ideally, ML algorithms should be able to use whichever features are necessary, discarding the irrelevant ones. However, to implement feature reduction as a preprocessing step could reduce the space and computation complexity, allowing to understand better the problem through data visualization. This can be achieved by employing two techniques: *feature selection* and *feature extraction*. In feature selection, the objective is to find the optimal number and combination of features, that means, the k most meaningful of the d dimensions and discard the other $(d - k)$ dimensions. Differentially, in feature extraction, the wanted result is a new set of k dimensions that are combinations of the original d dimensions [167], where the most employed method is the Principal Component Analysis (PCA) (detailed in Section 3.3.1.2).

3.3 Knowledge extraction

After completing the features and associated samples elaboration, data is almost arranged to be consumed by the ML algorithm, which is responsible for modeling the activity behaviors. To highlight that, before the knowledge extraction is performed, depending on the supervised ML algorithms, some extra and preprocessing steps are required. The recommended course of phases is illustrated in Figure 3.6. It shows all the adopted pipeline until the final goal is achieved: classify events as normal or anomaly; and the following sections will describe the represented stages.

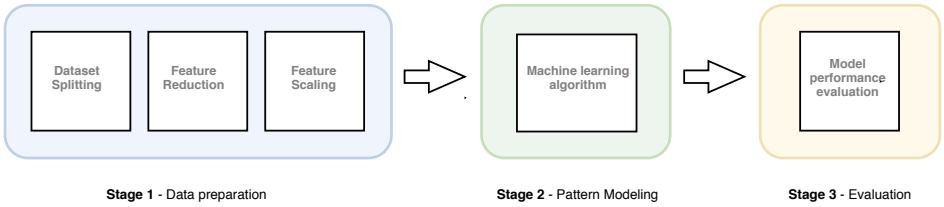

Figure 3.6: Description of data pipeline stages. Data begins by being transformed to be suitable for Machine-Learning algorithms. In the final phase is performed the Machine-Learning algorithms evaluation.

3.3.1 Data preparation

3.3.1.1 Data splitting

The first step of stage 1 is the division of the full dataset into training and test set. It is common to employ the *Pareto principle* where randomly is assigned 80% of the complete dataset for training and, therefore, 20% for the test set. This empowers a purely random split that typically does not introduce any issues when the dataset is large enough. Otherwise, sampling bias could be introduced. Along these lines, it is necessary to guarantee that data is divided into subgroups that are representative of the overall data. More precisely, the existing data category proportions in the full dataset should be maintained on the newly created subsets. This is known as *stratified sampling* and should always be taking into consideration for the division strategy.

Moreover, the data division concern does not occurs only once. As described in Section 2.5.3, cross-validation is used to evaluates the model's training accuracy, where the training set once again is divided into a smaller training set and a validation set. It creates a N distinct *folds*, training and evaluate the model N times, picking every time a different folder and train on the $N-1$ remaining folds. So, enhancing that even, in this case, the stratification should be safeguarded, preserving right classes proportion concerning the train and validation sets.

3.3.1.2 Feature reduction

As described in Section 3.2.5, fed an ML algorithm with all the event descriptors (i.e., features) without any previous analysis, will be translated into higher complexity and time consuming on prediction tasks. Feature reduction is mandatory to simplify classification and group correlated features. This adjustment may cause the loss of some information, although, it refines the noisy and redundant variables. Furthermore, the visualization is more straightforward, by breaking down into fewer features, enabling the discover relevant insights about activity patterns.

PCA is widely used as a dimensionality reduction method, consisting into a linear projection by firstly identifying the hyperplane that lies closest to the data, and then, projecting data onto a lower-dimensionality hyperplane. The projection has the following criteria: maximize the

variance and reduce the mean square distance between the original dataset and its projection. Choosing the projection with higher variance diminish the loss of information. Figure 3.7 shows a example of PCA reduction.

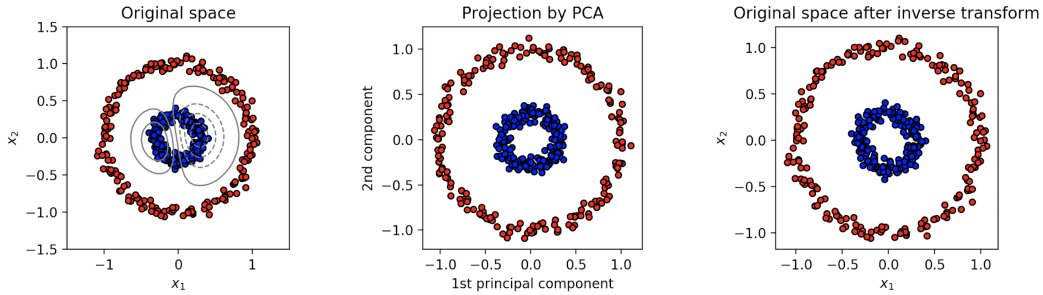

Figure 3.7: Example of PCA reduction over a fabricated dataset [168]. It is demonstrated that the inverse transformation validate that component reduction conserved all most information.

From left to right, the first image represents the original data deposition, the second image, in its turn, describes the PCA reduction over the original data where the best unique features that describe the sample are computed. The final image proves that no valuable information is lost by inverting the reduction process.

Furthermore, choosing the number of dimensions to reduce the dimensionality problem should not be arbitrary. Except for the visualization case, where generally two or three dimensions are handled, it is desirable to embrace the number of dimensions that stands for a sufficiently large portion of the variance (e.g., 95%). A simple procedure to verify and certificate these ideal variance percentage is to plot the explained variance as a function of the number of dimensions, as shown in Figure 3.8. Typically an elbow in the form of a curve is formed, indicating the point where the explained variance stops growing fast, also, helps to understand which number of dimensions has less loss of variance, and therefore, relevant information [126].

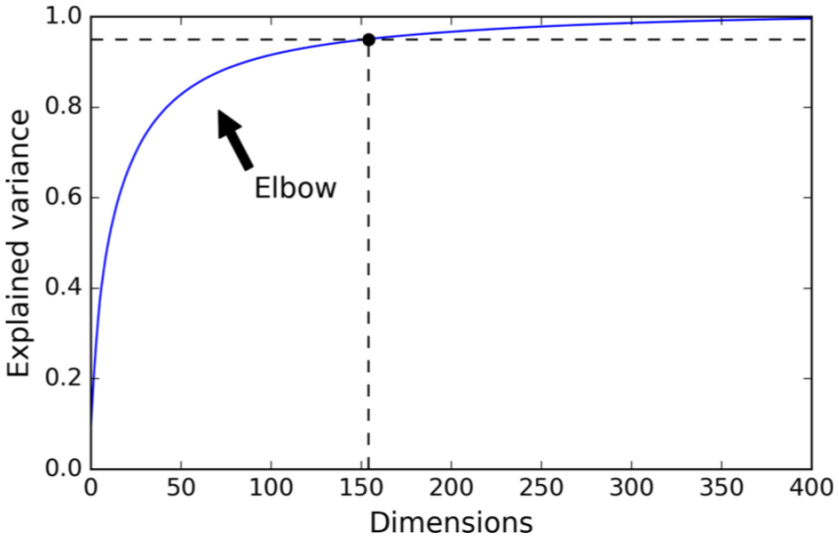

Figure 3.8: Example of an explained variance as a function of the number of dimensions [126]. It is possible to see that reducing the number of dimensions to 130, there will not be much loss of relevant information.

3.3.1.3 Feature scaling

The analysis obtained after the data overview, as elucidate in Section 3.2.4, can indicate the presence of numerical attributes with different scales and distributions. This last step of Data preparation's stage it is an essential transformation to not degrade predictive performance of ML algorithms. However, some algorithms are exceptions not requiring this preprocessing stage, as will be descried further in Section 3.3.2.

There are two transformation types to ensure that all attributes have the same scale:

- Normalization

  It shifts and bound the values to a specific range, $[0, 1]$ or $[-1, 1]$, by subtracting the min value and dividing by the max minus the min [126].
- Standardization

  In this case, there is no rescaling to a fixed range. The goal is to guarantee that each value has a zero mean (i.e., removing the mean) and unit variance distribution, subtracting each value by the mean and after, divided by the variance.

In ligth of this, Standardization, by not bounding values, could be a constraint to some algorithms like Neural Network (NN) that expect input numerical values ranging from 0 to 1. Moreover, it is much less affected by outliers. Considering a dataset with outliers, this values expose a different scale dimension in relation to the remained ones. Thus, by shrinking the range, outliers information will be lost, and their are crucial for modeling the activity pattern.

One other important concern, is that the employed scale should be the same both in the training and test sets, avoiding numeric data discrepancies.

3.3.2 Pattern modeling

In order to classify modeling network observations and predict an output, labeling the data is a demand. The essential objective is to classify between normal and anomalous network communications. This is the primary classification method, where only are two classes - *binary classification*. Yet, at a corporate level, network activity is not so simple to analyze, being a convoluted environment, where various malicious threats occur. In this way, aggregating different attacks behaviors on a single group could increase the problem complexity and reduce the capacity of distinguished anomalies dynamics and patterns, culminating into a poor generalization.

Furthermore, assuming the incursions diversity, one other possible approach is to assign a label to each attack vector, creating more than two different classes - *multiclass classification*. Like so, the classifier can learn easily and differentiate the characteristic patterns of each divergent communication. However, the predictive competence could be influenced by similar network observations belonging to distinct classes. So, the ML algorithms are seen as a big "black boxes" with N input classes outputting only a single label. This can be decomposed into smaller sequential "black boxes" that handle with binary classification problems. Breaking down the problem allows to have more control over the classifying course and endorse the application of different counter measures for each network irruption. There are two techniques for segregating a multiclass classification: *One-versus-Alls* and *One-versus-One*.

Assuming that is pretends to control three different attack vectors, for instance, a DoS incursion, a ransomware propagation, and a data exfiltration attacks - it is a multiclass problem formed by three classes. Using the *One-versus-Alls* strategy, three binary classifiers are built, being each one responsible for one class. When, new instance arises, each classifier provides a decision score for the specific instance, selecting the class whose binary classifier outputs with the highest score (winner-takes-all strategy). Contrarily, in the second strategy, each binary classifiers deals with a par of positive/negative class (i.e., between normal traffic and DoS attacks, between normal traffic and data exfiltration, and so on). Moreover, if there are N classes, $N(N-1)/2$ binary classifiers are generated, showing a fast class growth. For instance, in this situation with three classes will result in six binary "black boxes". At last, each binary classifier assigns the instance to one of the two classes, and the most voted class determines the instance classification.

3.3.2.1 Support Vector Machine

SVM is a classification algorithm, where the main idea is to find the boundaries between known samples, defining a *hyperplane*. Like so, SVM is sensitive to feature scales, being necessary that Data Preparation's stage (Figure 3.6) including feature standardization 3.3.1.3.

According to the dataset, data could be easily separated, creating a straight line and establishing a decision boundary with a maximum margin (corresponding to the Linear SVM case). Figure 3.9 exemplifies the described separation, where the decision boundary is supported by the instances placed on the edge, the *support vectors*.

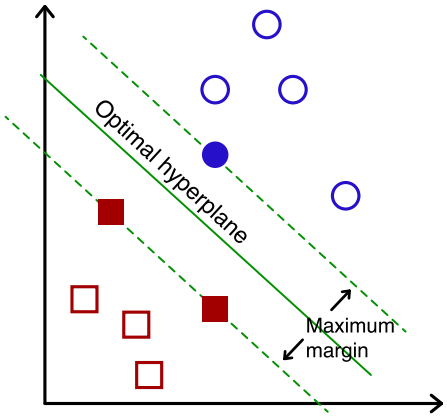

Figure 3.9: Exemplification of the SVM's strategy for a two-class problem where the instances of the classes are shown by squares and dots. The thick line is the optimal hyperplane and the dashed lines define the margins on either side. The fill instances are the support vectors.

Finding a good balance between keeping the margins large as possible and limiting the violations (i.e., instances that end in the middle of the boundary decision or on the wrong side [126]) is a challenge for well performed classifications. This equilibrium is called *soft margin classification* and it is controlled by the *C parameter*. Figure 3.10 represents C parameter influence in the decision boundary shape.

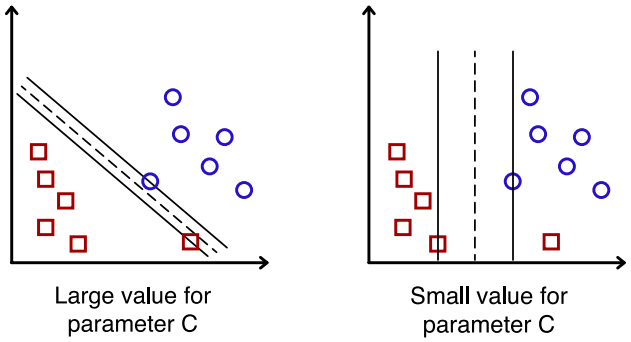

Figure 3.10: Visualization of the impact of SVM's C parameter in the separation margin and the tolerance in relation to outliers created by such separation.

A large C value models a minor decision boundary, but with less violations, as the left graphic of Figure 3.10 exposes, presenting a better generalization for outliers. Instead, smaller

values for parameter C produce wide margins, as it is visible on the right side of Figure 3.10, however with more violations.

Many datasets are not linearly separable, as the case of heterogeneous network data, and in this circumstances, SVM is considered a Nonlinear classifier. In nonlinear problems, the data is mapped to a new high dimensional space, where the formed linear model in the new space corresponds to a nonlinear model in the original space. This is possible due to the *kernel tricky* [167].

Training SVM, labeled data is needed and, originally, is a binary classifier. In this away, to cover multi-class problems, SVM algorithm empowers *One-versus-all* and *One-versus-One* strategies (explained in the above section). Likewise, SVM also could be extended as an unsupervised learning algorithm which tries to separate all data without having labeled data, the *One-class* SVM [169], being applied and implemented on outlier detection problems [167], [170].

3.3.2.2 Decision Trees

DT is also a ML algorithm capable of performing classification, characterized by a hierarchical structure and composed by the following main elements:

- *Root node*
- *Decision node*, which applies a test function to an attribute
- *Branch*, that represents an outcome of the test to an attribute
- *Leaf node*, representing the final object class label

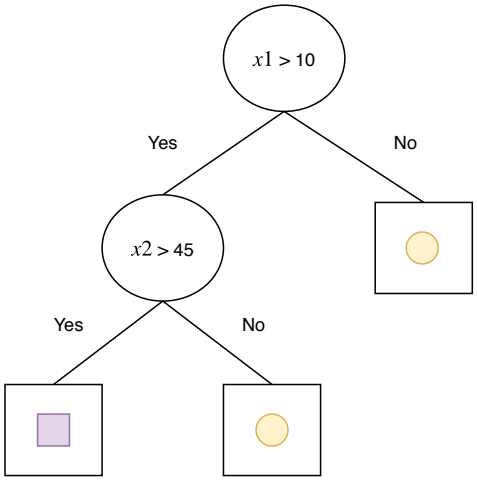

Figure 3.11: Example of a DT structure. Oval nodes are the decision nodes and the rectangles are leaf nodes.

Using Figure 3.11 to understand how DT works, it starts at the root node, where each decision node applies a test function, tagging the outcome with a label in the branches. Along the tree, tests are enforced doing recursive splits, until the leaf nodes are hit, giving the final label. The quality of a split depends on the *impurity measure*. A split is considered pure if after the division all the instances choose a branch composed of items of the same

class. In other words, the split objective is to form homogeneous groups of the training set. Thus, when a pure split is reached, divisions stopped and label can be assigned to the leaf node. The function that calculates the impurity is defined as *entropy*, wherein each step during tree growing is chosen the split that causes entropy values close to zero. In this way, DT implements a divide-and-conquer strategy, broken complex functions into small and more manageable problems. Due to its easy interpretation, DT is very popular, many times preferable over more accurate methods [167], and highly used in anomaly activity detection [171], [172].

3.3.2.3 Ensemble Classifiers

All the previous elucidated classification algorithms are working solely over the categorization data problem, that means, the classification outcome is based on the most accurate result provided by a single predictor. *Ensemble* method emerged to deal with complex tasks, building different types of approaches to solving the same problem. In ML, the basic idea is to gather a group of predictors (called an *ensemble*) to improve the accuracy over a single estimator. Ensemble learning has two main techniques:

- **Bagging:**

  Every predictor is trained with the same algorithm, but implementing a *bootstrapping* technique. That means each predictor is trained with a different random subset of the original training set with replacement, making possible the same instance appears in different subsets (bagging). The classifiers are used in parallel, as described in Figure 3.12, allowing training instances to be sampled several times. When facing a new instance, each predictor classifies it individually, being the final classification decision based on the class with the majority of votes.

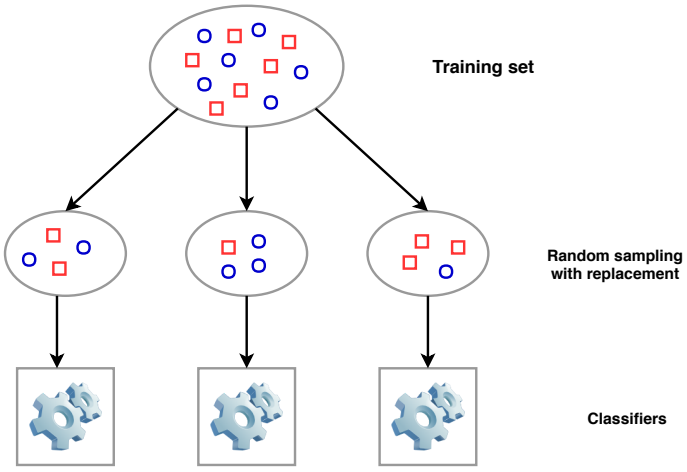

Figure 3.12: Representation of the Bagging technique. Predictors are able to act in parallel due to random sampling with replacement of the original training set.

- **Boosting:**

  It combines a set of weak learns to create a final strong learner, where iteratively, each new classifier is influenced by the previously built ones, trying to correct their

misclassified predictions, as Figure 3.13 illustrates. Adversely to the bagging strategy, where the predictors act in parallel, regarding to boosting, each predictor depends on the previous one, not scaling as well as bagging, once it cannot be parallelized.

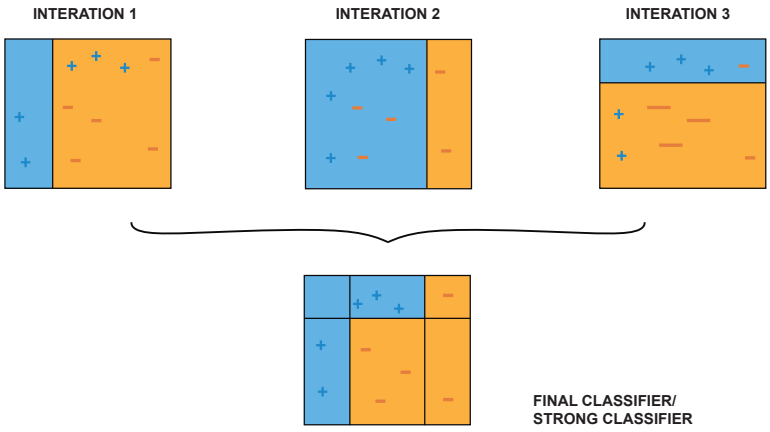

Figure 3.13: Representation of the boosting technique empowered by ensemble methods, iterating and self adjusting the decision edges, until the desired number of predictors is reached or when a strong classifier is founded.

Random Forest is an ensemble of decision trees that uses bagging for training. Moreover, there are two central boosting implementations, *Adaboost* and *Gradient Boosting*. Besides, recently a new implementation *Xgboost*, became widely used in Kaggle competitions having already won several prizes [173]. *Adaboost* algorithm applies boosting techniques and acts by creating several weak learners, where the new predictor corrects its predecessor by increase and update the relative weight of misclassified training instances, in order to give more emphasis to the training instances misclassified and build a strong classifier. *Gradient Boosting* acts in the same way, however, instead of adjusting the instance weights at every iteration, this method fits the new predictor based on the residual errors made by the previous predictor [126].

Further, usually, ensemble algorithms use decision trees as base predictors making them invariant to feature scaling, being scaling step skipped when dealing with one of this methods.

3.3.2.4 Neural Networks

When facing with network monitoring problems, systems usually deal with large-scale datasets. SVM despite being an effective method of intrusion classification, is typically a quadratic optimization problem [174], that will decrease its performance when handling extensive datasets. Moreover, data generally comes in a streaming fashion, and SVM does not support online learning.

So, NN techniques, the main concept of Deep Learning, have gained popularity and successfully applied to fields like speech (e.g., Google's Cloud Speech) and music/audio signal recognition(e.g., Shazam). To solve the drawbacks described above, different NN implementations have been tested in network pattern identification problems [175], [176],

making a IDS more scalable and able to solve complex ML tasks. *Perceptron* [2] is the simplest architecture of NN, composed by multiple inputs and one output with associated weights. MLP combines multi-layers of perceptrons, characterized by *input, output and hidden layers*, as represented in Figure 3.14. In this implementation, the network could have or not a *bias neuron* and a *backpropagation* algorithm. After the network has an output value for each neuron, it calculates the output error, checking the difference between the desired output and the actual output. Then it reviews how each neuron in the previously hidden layers contributes to the error until reaching the input layer. The main idea behind the backpropagation is to slightly change the weights until having a result closer than expected. So, NN in training uses Stochastic Gradient Descent to optimize the weights according to a specific solver [126].

In the same way that SVM, NN also is sensitive to feature scale, and therefore the pipeline requires feature scaling (3.3.1.3), converting each attribute to the scale with a mean of zero. Moreover, to accomplish better results, it is necessary tuning a number of hyperparameters such as the number of hidden neurons and layers.

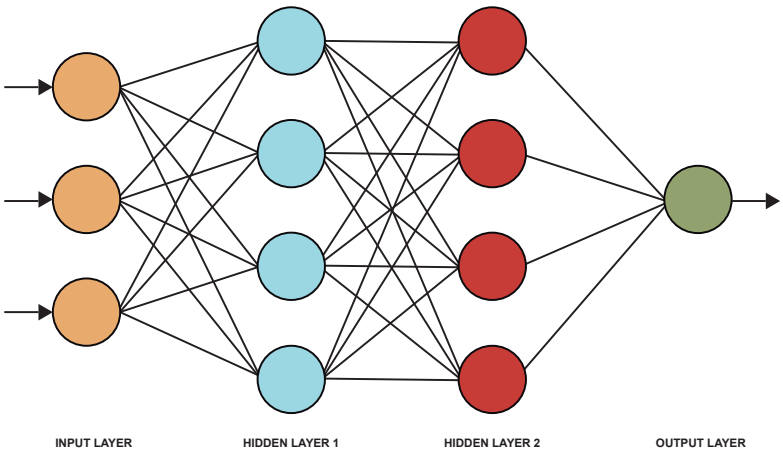

Figure 3.14: Neural network schema composed by the input and output layers and two hidden layers. In case of having two or more hidden layer, neural networks are designated as Deep Neural Network.

3.3.2.5 Novelty and outlier detection

The performance of ML classification algorithms depends on the provided data quality, and most of real world datasets do not contain meaningful examples of abnormal samples. In this way, a good solution to contradict this lack of negative examples and performing anomaly detection is outliers detection. The main idea is to separate the regular observations from some polluting ones, the outliers. Outlier is an observation which deviates from the other observations appearing to be incompatible. Likewise, novelty detection has the same goal, yet outliers do not constitute the training data. For once, in novelty detection the outliers are

[2] Also called linear threshold unit (LTU)

discovered in new observations, contrarily to outlier detection, that needs to fit the central mode of the training data, ignoring the deviant observations. On both methods, the normality distribution and borders are learned from the normal instances creating a base model for that population. Thus, when a new instance arrives, if it lays outside the learned frontier it is classified as an outlier sample. Otherwise, is seen as an inline observation similar to the existing ones.

The novelty strategy is demonstrated in Figure 3.15, where is implemented the One-Class SVM algorithm [169] over some random data, to perceive how the decision function build the frontier and classify new data observations.

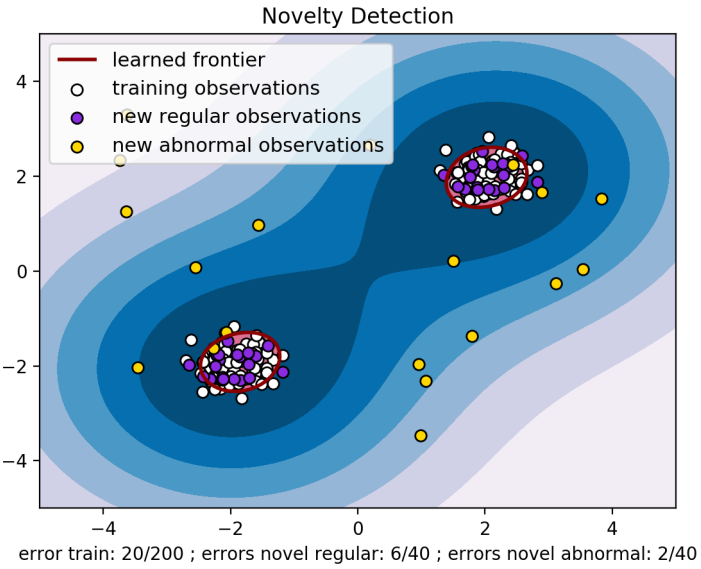

Figure 3.15: Novelty detection example over random data using One-Class Support Vector Machine showing the learned frontier and the anomaly classification errors [177].

3.3.3 Performance evaluation

In the case of binary classification problems, some measures that reveal the model correctness facing its predictions, in other words, metrics that tell the model performance. Focusing on a binary classifier existing four possible predictions for a two-class problem:

- *false positive*

  When the outcome is incorrectly classified as positive, when in fact belongs to a negative example.
- *false negative*

  When the classifier labeled as negative contrarily to the positive true class.
- *true positive* and *true negative* representing the correct classifications

Together they composed the *confusion matrix*, as described in Figure 3.16, which summarizes the prediction results of the classifier. This is the base to empower more complex performance measures.

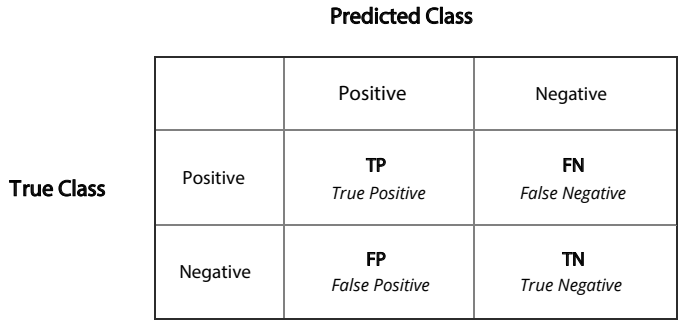

Figure 3.16: Confusion matrix for a binary classifier, where the columns represents the classifier prediction and the rows are the actual classes. This also can be empower in multiclass problems, where the matrix grows proportionally.

The most common performance metric is the *accuracy*. It represents the fraction of examples correctly classified. Looking to the confusion matrix is simply the sum of diagonals divided by the total, forming the following equation 3.1:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

Despite accuracy being a straightforward metric, the first moment of evaluation should focus on cross-validation (elucidated in Section 2.5.3). The model in each iteration is trained with a training set's percentage and tested against the test fold. This iterative process, not only avoids biasing but also is useful to reveal the model's learning behavioral by analyzing various accuracies obtained at the end of each fold. For instance, cross-validation can elucidate the training accuracy and the average of all accuracies, crucial to detect overfitting. Withal, most real world datasets are unbalanced [3] and *accuracy* treats all examples as the same, not taking into account the class proportion. Consequently, the majority class tends to have high accuracy and the minority class, in its turn, smaller accuracy percentage. Thus, cross-validation also is a good tool to overcome this problem, along with the implementation of other metrics to computed the overall efficiency:

- *Specificity*

  Expresses the fraction of negative examples properly classified, i.e., true negative rate (equation 3.2)

$$Specificity = \frac{TN}{TN + FP} \tag{3.2}$$

- *Recall* [4]

[3] The number of positive events are not equal to the number of negative events.
[4] Or *sensitivity*

Antagonistically to specificity, recall refers to the ratio of positive classifications correctly classified - true positive rate (equation 3.3)

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

- *Precision*

  Traduces the accuracy of the positive predictions. In other words, symbolizes the current fraction of positive samples in the group that the classifier assigns as positive (equation 3.4)

$$Precision = \frac{TP}{TP + FP} \tag{3.4}$$

There is a direct correlation between *precision* and *recall*, establishing the *precision/recall tradeoff*. This tradeoff indicates that a classifier can not have elevated values of *precision* and *recall* at the same time: increasing precision reduces recall and vice versa. So, it helps to balance between the extremes, being the calibration dependent on the objective of the system. As the context of this work is to detect an anomaly, a system with high precision and low recall are preferred, predicting positive only when very sure, instead of predicting everything as positive.

Also, these performance metrics are also related to the *f-measure* (described by Equation 3.5). It is called as the harmonic mean of *precision* and *recall* [126] because gives more weight to low values, instead of treating equally all values, and for that reason, it is a good way for comparing two classifiers.

$$F - measure = \frac{2 * Recall * Precison}{Recall + Precison} \tag{3.5}$$

confusion matrix should always be taken in attention, as it is the core for execute all this evaluation measures. Moreover, *ROC curve* is another way to evaluate a classifier. It is very similar to the *precision/recall tradeoff*, but ROC curve designs the *true positive rate* against the *false positive rate* [5]. Comparing various ROC curves performed by different ML algorithms empowers a straightforward way to compare the classifiers performance in detection of anomalies.

[5] The ratio of negative instances that are incorrectly classified as positive, and it is equal to 1− the true negative rate, i.e., 1− *specificity* [126]

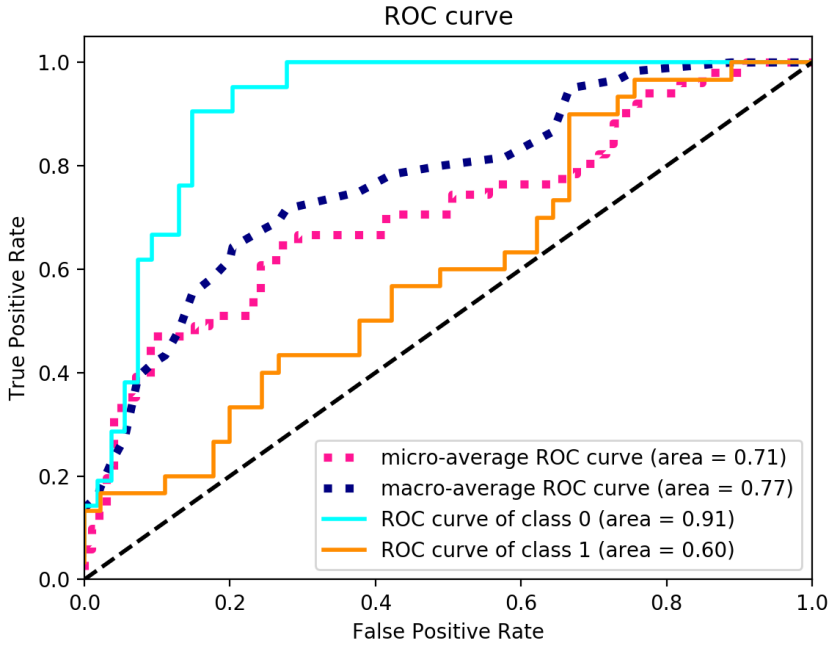

Figure 3.17: ROC curve for binary SVM classification using the IRIS dataset [178]. It feature true positive rate on the Y axis, and false positive rate on the X axis.

Once again there is a tradeoff: the higher the recall (corresponding to TPR), the more false positives (FPR) the classifier produces. As represented in Figure 3.17, the dotted line represents the ROC curve of a purely random classifier, for which a good classifier should be far. The optimal point is the top left corner where false positive rate is zero and the true positive rate is one. Furthermore, measure Area Under the Curve (AUC) permits to compare classifiers. A perfect classifier will have a ROC AUC equal to 1, contrarily, a purely random classifier will have a ROC AUC equal to 0.5 (similar to the ROC AUC for the class 1 illustrated in Figure 3.17).

3.4 Conclusion

In conclusion, although each corporate network environment has its own characteristics making each case unique, the described methodology in this chapter, is a generic process that can be extended for other dissimilar and more complex scenarios. It starts by monitoring and gathering network data, passing by the feature engineering and ending with the knowledge extraction. It is mainly designed to build an inner-network communication profile and detect outliers behaviors by actively monitoring the network and employing machine learning techniques.

CHAPTER 4

Proof of Concept Scenario and Results

Tis chapter describes the used scenario, along with the explanation of how network data is created. A briefly overview of the dataset is given and the described ML algorithms of this dissertation are tested, applying the proposed evaluation metrics. The adoption of actual and suitable attack vectors to generate and analyze the ML anomaly detection capability, marks this proof of concept.

4.1 Network data generation

As discussed in Chapter 2 of this dissertation aims to detect propagation movements and data leakage in a corporate environment. Intending to be more similar as possible from an organization network domain, a network scenario was designed in GNS3 [1] which was used to simulate the propagation and data exfiltration incursions. Evidently that this topology is far from a real corporate network, however, was built taking into account the architectural principles explained in Section 2.1.1. It has three layers - Access, Distribution, and Core; being implemented VLAN segmentation at access layer (as shown in Figure 4.1) and are constituted by the following machines:

- VLAN 1:
  - win7-2: Windows 7 machine.
- VLAN 2:
  - win7-1: Windows 7 machine.
- VLAN 3:
  - win7-3 and win7-4: Windows 7 machines
  - ubuntu-1: Ubunutu machine that represents the contaminated enterprise host.

[1] Network software simulator

The focus of this work is to deal with a scenario where a machine is already infected and compromised, and not targeting the previous moment before the corruption. This is the starting point of the detection. Therefore, to build a system that learns form network observations to find deviations from the normal behavior, it is required the creation of propagation and data leakage attack vectors to "fabricate" the network data that will be provided to the ML models. So, this matches with a classification problem (supervised learning) as argued in Section 2.5.3, is necessary to assemble network examples for normal and uncommon traffic. In sections 4.1.2 and 4.1.1 will be explained the data network process creation for propagation and data exfiltration cases, respectively.

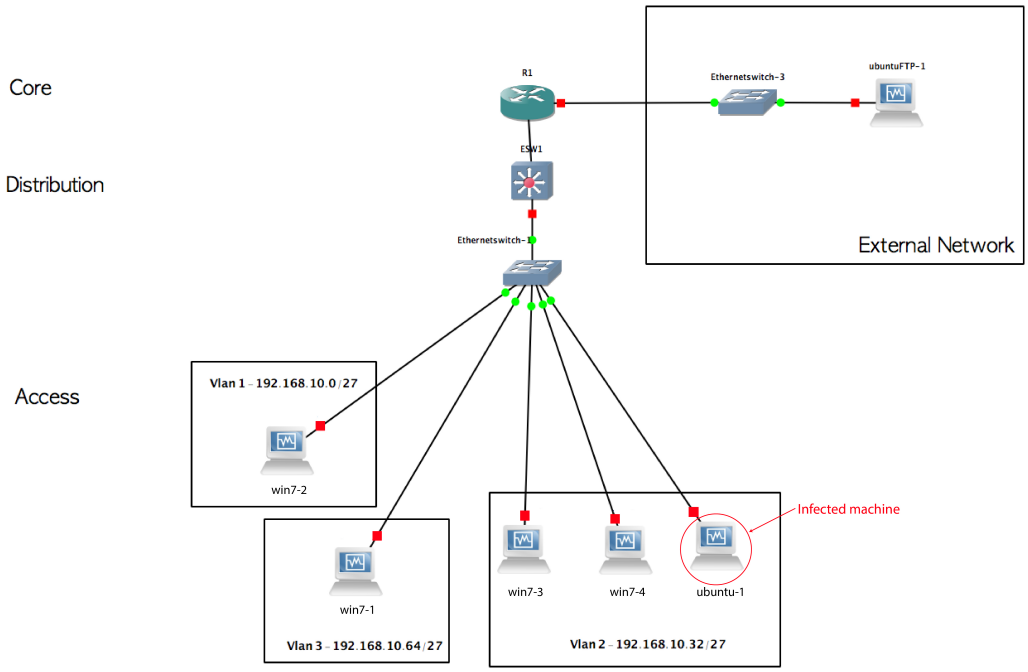

Figure 4.1: Network topology for a controlled simulation. It has organized into Access, Distribution and Core layers. The access layer represents the internal corporate network, which is structured by three distinctly VLAN. In right side is represented the external network, outside of corporation borders. This is a simple and fully connected network only to simulation intent, not having been considerate the redundancy and resilience aspects.

4.1.1 Propagation data production

4.1.1.1 Abnormal activity

Wannacry and NotPetya ransomwares were the major attacks that start to benefit from the NSA exploitation tools, as already argued in Section 2.3. Both use Eternalblue exploit for lateral movement, and in this work, was pretended that propagation attack was identical as close as possible to the spreading method of these attacks.

Eternalblue [2] targets machines prior to some Windows 7 versions [84]. These machines contain an interprocess communication share (IPC$) that allows a null session. That means,

[2] CVE-2017-0144 vulnerability in MS17-010

the connection is established via anonymous login and a null session is permitted by default, approving the client to send different commands to the server (notice that SMB has a client-server architecture 2.1.2). This is one of Eternalblue's bugs and vulnerabilities that are elucidated by CheckPoint's researches in [179] revealing the SMB exposure. In this way, a Thai security researcher, Worawit Wang [3] implemented a proof of concept of MS17-010 vulnerabilities, making the code available in their github repository [180]. The Wang's repository referenced by [181], [182] was used to carry out the Eternablue exploit in this dissertation. However, some changes were made to the original version. Worawit's Eternalblue exploit only targets and infects one specific machine defined by an introduced IP address. To simulate a lateral movement in the network was added some extra functionalities. It will be described below by order of events:

1. Depending of the wanted network range, the script performs a scan over all IP addresses in the defined network mask. Thus, the infected machine starts sending ping commands to the IP addresses to check the active machines.
2. After found the up hosts, verifies if the target is MS17-010 patched or not, using an auxiliary script provided by Worawit. In the positive case, the infection proceeds.
3. Then, it is used the port scanner of nmap to validate if the machine has the TCP 445 port opened.
4. Having all victims, the script starts to launch the Eternablue exploit sequentially to all machines, spreading across the LAN.

Furthermore, to make the attack more realistic and closer to the human behavior, the inactivity time is defined by random numbers from the exponential distribution, where the mean parameter is changed after every attempt to infect a machine. This procedure aims to increase the generated data quality, establishing an activity frequency antagonistic to a bot or machine (which acts periodically), and forming a more robust attack vector capable of challenging the ML classifiers performance. Along with these lines, three types of attack mode were conceived. So, thinking in a scale of action's speed, two of them are in opposite ends (the faster and the slowest extremes), and the third, in a medium-term between them. Regarding the quicker fashion, the network scan is performed over a reduced number of IP addresses, and it is characterized by performing an intrusive outbreak. Diversely, the slowest one carries out a deep scan over the LAN and the exploit launching takes a careful and discrete action. In its turn, the medium type empowers an intermediate level between an aggressive and cautious disruption. Just so, infections were launched among the three VLANs, being their ferocity defined by the mean values used to produce the silence period, where high mean values form long periods of inactivity, shaping a more human style.

Notice that, for security questions, all this infecting process were done in the GNS3 network (Figure 4.1) without any connection to the Internet. Also, as the goal is to generate network traffic after the machine was compromised, the internal malware execution is out of the scope of this dissertation, therefore was not performed.

[3] Designated as _sleepya on Twitter

4.1.1.2 Normal activity

Once will be utilized classification ML algorithms, it is needed the creation of network traffic that represents the normal action of machines in the network. In the corporate simulation network, each VLAN is composed of machines with Windows 7 operation system. As seen in Section 2.1.2, SMB is the protocol used by Windows machines for providing shared access to files between nodes on a network. Following this lines, machine win7-2 has a shared folder with all the others Windows machines (win7-1, win7-3 and win7-4). Moreover, to produce natural behavior, regular file and directory actions like upload, delete and copy, were made by each machine over the created shared folder. At a network level, this activity is translated into TCP traffic for port 445, being the exchanged data volume conditioned by the file/directory size. This way, the three presented actions can be established using distinct dynamics, that are responsible for regulating either the files/directories size as the activity times, which delineated the action signature. Once again the silence periods are described by random numbers with an exponential distribution, in order to formulate network observations identical to those produced by a human user. Also, in spite of not create a repetitive pattern concerning the volume of bytes transferred in the internal network, file size variance was introduced by executing actions that deal with different file size magnitudes: KB, MB, and GB.

4.1.2 Exfiltration data production

At an organizational level, as explained in Section 2.1, there are policies that define what services are allowed or not. In this Proof of Concept, relatively to the exfiltration case, was assumed that only the 22, 80 and 443 TCP ports are tolerated by the company, corresponding to SSH and file transfers (e.g., SCP), HTTP and HTTPS protocols, respectively. In this way, was elaborated leakage attacks using these three protocols for forwarding data outside of enterprise borders, but first will be elucidated how normal traffic was established.

4.1.2.1 Normal activity

The method used to find the normal activity profile was the same for SCP, HTTP, and HTTPS protocols, albeit, for simplicity on explanation, will be elucidated in detail for the HTTPS case. Being the previous ports allowed by the enterprise, corporation users can then use several services that run over those protocols for transfer and share files over the network. Over a while, arbitrary uploads of files with different sizes were performed (from 100 KB to 500 MB) for Google Drive [4] along with usual browsing (social networks and googling). These actions aim to represent the typical habits of a regular company user. From the network observations caused by these actions, it was plotted a graphic that displays the number of upload bytes per second, to understand their network behavior. The result of this process is represented in Figure 4.2.

[4]File storage and synchronization service developed by Google that enables synchronization among different devices and file sharing operations

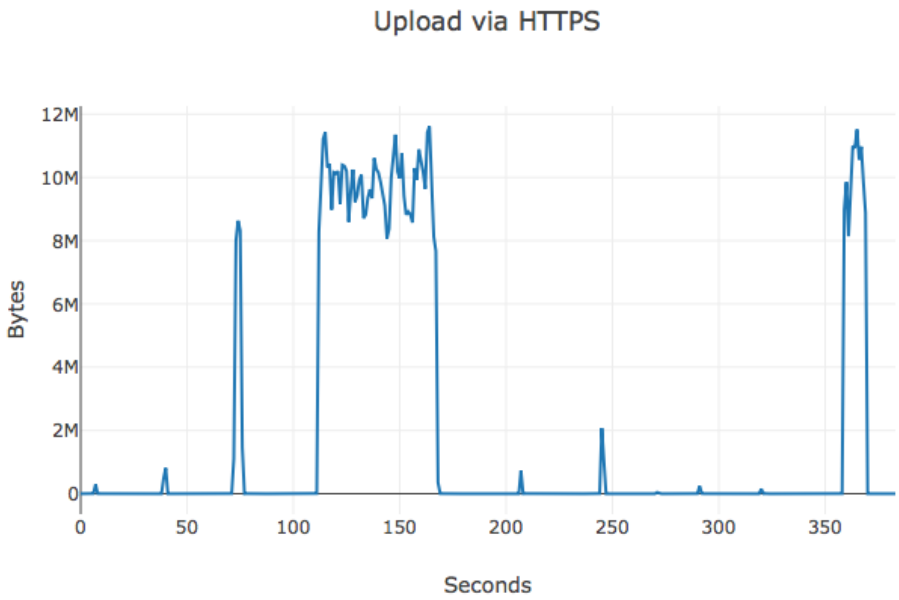

Figure 4.2: Plotted graphic that illustrates the relation of number of bytes transfered per second, collected from a sample of the upload activity made to Google Drive

Therefore, from graphic analysis the following features were extracted to trace the profile of HTTPS normal activity:

- **Peaks amplitude (PK):** The amplitude, that is, the y-value, representing the maximum number of bytes that can be sent. It acts as a threshold of bytes.
- **Peaks duration (PD):** This shows the period of activity, i.e., how long the user was uploading files.
- **Interval between peaks (SP):** The interval between peaks, contrarily to the activity period, traduces the inactivity time, the silence periods.

These three metrics were organized on a tuple where each entry represents the associated feature - [PK, PD, SP] [5]. Considering the total time of activity, several tuples were formatted and, posteriorly, handled by the script responsible for simulating the obtained profile. In such way, was used the Google Drive API to perform uploads of files without human supervision on the following mode: every time that an upload is executed, one of the tuples is randomly selected, assuming a discrete probability distribution, to get the maximum of bytes allowed, the time of activity and silence period. Surely, this process is influenced by the initial sequence of files uploaded and by the network performance, but the goal is to build a more reliable normal profile close to human behavior. To point out once more, that the same examination was shaped by the other protocols using an equal range of file sizes, but instead of being used the Google Drive API to perform uploads, regarding to HTTP case, was empowered the Requests's python library [183] to execute GET and POST demands to several sites that still

[5] Where PK corresponds to the peak amplitude, PD the peak duration and SP the silence period

not had the newest security feature implemented by the SSL/TLS protocol. Regarding the SCP, the file synchronization was done using the scp's python library [184].

4.1.2.2 Abnormal activity

To carry out the data leakage outbreak it was assumed that the ubuntu-1 machine in the simulation network environment is contaminated. After the infection, it could be controlled to sent confidential data to the exterior of the internal network. In this fashion, as already said, in this Proof of Concept, the allowed and, supposed, trustworthy enterprise protocols were used to mask the data exfiltration movement. So, similarly to the data propagation production, is intended that all data exfiltration actions be, as far as possible, unrelated to a behavior addressed by a machine. In such manner, the activity time is always set to a different value comprised in an exponential distribution, ensuring that are not periodic actions. All the dynamics conceived for the three protocols (which will be described throughout this section) were elaborated taking into account the normal profile associated with each one, varying the sent volume of data send and the period of inactivity.

Beginning by the anomalous SCP attack vector, it was simulated on GNS3, using the machine placed on the external network (Figure 4.1) representing the outside attacker's machine. Thus, through a file transfer script (using scp's python library [184]), ubuntu-1 host forwards data to the external peer, using one of the six different dynamics described in Table 4.1.

Table 4.1: Description of the empowered dynamics to leak data over SCP channel, showing from stealthy dynamics to more intrusive ones.

Dynamic	File size range (KB)	Mean inactivity maximum value (seconds)
1	6 - 60	50
2	10 - 30	60
3	70 - 500	100
4	80 - 10240	200
5	900 - 3072	150
6	10240 - 1048576	250

So, the above SCP's dynamics are characterized by two variables, the upper and lower size limits of the data leaked and the maximum number that can be assigned to the exponential distribution mean parameter. More precisely, the mean inactivity maximum value delineates the inaction duration, where a random number (limited by the maximum established) are attributed to the mean, making it always different. The combination of these two metrics intents to define the intrusion degree of an attack, where large sizes are followed by high mean values (i.e., more extended periods of silence). For instance, dynamic 1, characterized by small data volume and mean value, performs a more intrusive leakage than the dynamic 6, which deals with Mb and higher silence values, assuming a more restrained action mode.

Moreover, regarding HTTP and HTTPS, the external machine in GNS3 was not used. Instead, the ubuntu-1 host already has Internet connection available, and it can communicate with external IP addresses and services. To leak information using the HTTP protocol

were sent elementary POST [6] carrying data to websites that still do not support SSL/TLS encryption. In the same way that SCP abnormal activity, was created six HTTP dynamics shown in Table 4.2.

Table 4.2: Summary of the distinct dynamics created to perform poisonous POST (via HTTP channel) containing confidential information transfered to the company outside.

Dynamic	File size range (KB)	Mean inactivity maximum value (seconds)
1	10 - 40	40
2	20 - 80	80
3	70 - 800	100
4	100 - 1024	150
5	900 - 3072	150
6	10240 - 1048576	200

Same as the SCP's dynamics table defined in Table 4.1, the above HTTP's dynamics are ruled by the same goal: make incursions more and less stealthy and shaping a non-periodic distribution.

Steganography, the practice of concealing a file (such a text file, image or video) into another file, as shown in Section 2.3.5, is a way to camouflage confidential information or malware code inside a trivial image (or in a video file) being unnoticed by the human eye. In this dissertation, this technique was employed to simulate data exfiltration, since it is one method practiced and increasingly used by real attackers [185] that goes unnoticed by the security alarms, making the attack vector more realistic and closer to being a real incursion. In this manner, to put it into practice, it was used the nukeop's github repository [186] that implements a steganography's program, encoding files into images. The program transforms the wanted file into bytes that are separated into 2-bit sequences. Since every pixel has four channels (RGBA), and each channel uses 8 bits for its value, in the case of having an image with the following dimension, $1764 * 1797$, it is possible to hide a file with size less than or equal to 12,6 MB. Moreover, before being applied the steganography's process, the file is encoded to base64 (converted to binary), to decrease its size and producing a final image indistinguishable from the original one. Hosting the ultimate image metadata information just need to send the data. Being the traffic to/from TCP 443 port allowed, the poisonous images are transmitted and posted on Twitter (that uses HTTPS for secure communication over the Internet). So, what seems to be normal tweets are, in fact, a data leaked outbreak. In order to perform the images uploads was used the Twitter's API through a python wrapper - Twython [186]. Although, as described on Twitter documentation, it restricts the image size to 5 MB and only 2,400 tweets per day. Due to these limitations, the image size that would be applied to the steganography process was carefully chosen, conditioning the volume of data handling by the attacks dynamics. Along these lines, only two dynamics were fabricated, as described in Table 4.3, where the dynamic 1 is more invasive than the dynamic 2.

[6] Using the Requests's python library

Table 4.3: Definition of the steganography's attack vector dynamics that hide private company data inside a normal image and publish in Twitter by taking advantage of the HTTPS encryption features.

Dynamic	File size range (KB)	Mean inactivity maximum value (seconds)
1	0 - 10	60
2	50 - 600	100

4.2 Network Data Collection and Processing

As discussed in Section 3.1, the installation of a sniffer on every endpoint probably is undesirable. Thus, the place where it is allocated is decisive. Being dependent of the problem in question and looking to the existing monitoring tools (discussed in Section 2.4), in this case of study, the network data was captured at the switch level, making possible the aggregation of all data that flows on a LAN. Hence, Wireshark software was used in GNS3 for sniffing the switch interface connected to the distribution layer (Figure 4.1). In case the machine has Internet connection. Instead, it was deployed a sniffer on the host, capturing all data in the wireless interface. In both sniffer implementations, the captured raw data was stored on PCAP files. Therefore, in the scope of this dissertation, sniffers do not have any processing capabilities. That means the PCAP capture grasping by the sniffer is composed of raw data. So, it is required to transform raw data into meaningful statistical information. Also, if the ML model took raw data as input, it will increase the complexity of the ML algorithm, generating worse result accomplished with prolonged calculations.

4.2.1 Parsing process

After collecting and save network data traffic on a PCAP file, packets must to be parsed for extract meaningful information. As described in Section 2.4 most of the network traffic are abide by TCP, UDP and ICMP protocols, and in light of this work, network data will be filtered by TCP and UDP transport protocols, take the coming attributes from the 1 to the 4 layers of the OSI model:

- Layer 1 to 3:
  - Source and destination IP addresses
  - Volume of bytes
  - Unix epoch time
  - VLAN ID
- Layer 4:
  - TCP
    - source and destinations ports
    - SYN and ACK flags
  - UDP
    - source and destinations ports

For each dissected packet it is possible to know the packet direction (internal or external), once the range of a private network is defined by the VLAN 1 to 3 subnetworks (and the subsequent packet will be associated by the VLAN ID), being all the rest considered public network. In this way, packet attributes and the activity time (the difference between the timestamp of the first and last packet) are used and consumed to build the network attributes, composing the next step - the transformation of packets into samples 4.2.2.

Moreover, it is important to have in consideration that with a large amount of collected data in network corporations, and ambitioned a near real-time system detection, the performance of parsing process is crucial. Thus, in order to achieve faster processing and scalability, *Kaitai Struct* [187] was used for packet dissection. a C++ compiler forms it, surpassed any other python dissection framework, being able to process on average 31.00 packets per second, when used Python version 3.5.1 [188].

4.2.2 Converting packets into samples

In Section 3.2 it was generically explained a possible process to extract network observations from raw data, that in this work, was addressed as a recipe for converting packets into features. Thus, in the first step, described in 3.2.1, counting metrics are computed over a sampling window. Notice that, this is only the initial step, where the samples still do not represent network observations, being necessary advance to the next stage, 3.2.2, where features are defined and ready to be used in order to build a profile.

Therefore, propagation infection and data exfiltration incursions are two dissimilar situations, and for this reason, each one had a feature extraction process based on its characteristics, which will be described separately.

4.2.2.1 Propagation case

For the lateral movement threat, the formed attributes from the applied statistical processing over the metadata are described in Table 4.4 bellow:

Table 4.4: Low level description of the attributes defined in each sampling window for the propagation scenario considering now a creation of a more generic system

Sample attributes	Description
ip_internal	Number of internal IP addresses contacted
internal_bytes_up	Total number of internal bytes uploaded
internal_bytes_down	Total number of internal bytes downloaded
packet_count	Number of packets transmitted and received
tcp_sessions	Number of established TCP sessions

The normal behavior of a spreading force is to start contacting atypical and unusual machines, verifying an increase of observed IP addresses along with the variance. This is the reason for the ip_internal choose. Furthermore, internal_bytes_up and internal_bytes_down attributes traduce the volume of data uploaded/downloaded on the internal network, that

is, the sum of exchanged data among VLAN 1, VLAN 2 and VLAN 3. Both are used to expressed modifications of the internal pattern volume. For instance, a ransomware charge, after being dispersed on the network and gained access to enterprise NAS/SAN, will start to encrypt all the documents, provoking a hike of internal data volume. Besides, it provides a way to control changes from the upload/download rate of end hosts. Moreover, since it was utilized the SMB vulnerability to fabricate the propagation action (4.1.1), a good strategy goes through counting the TCP opened sessions to port 445. Firstly, because SMB file sharing protocol, that is being used by the Eternalblue runs over TCP on port 445. Secondly, by observation of the network behavioral produced by the attacker vector, it was verified that the Eternalblue exploitation establishes multiples TCP session over the time. However, to make the system more generic, in order to be applicable in different anomalous network situations and not be dependent of an attack vector, instead of controlling the number of TCP sessions for a specific port (the 445 port), this Proof of Concept is counting the overall TCP sessions. In this way, the attributes selection are less depending on the attack vector action mode.

It is essential to clarify how the number of TCP sessions was counted. TCP is connection-oriented, that is, before any data can be transmitted, a reliable connection must be obtained and acknowledged. This represents the TCP three-way handshake. For example, machine win7-2 wants to talk with win7-1 machine, so TCP creates a source port that interacts with a well-known port. Only after the three messages SYN, SYN-ACK, ACK have been transmitted (as demonstrated in Figure 4.3)the TCP session is established and win7-2 is allowed to communicate with win7-1. As discussed in Section 3.2, packets can be aggregated by the source IP address and the destination port, forming the notion of data stream aggregation. In this way, every time that an ACK flag is verified for a specific data stream, the number of the TCP session is incremented.

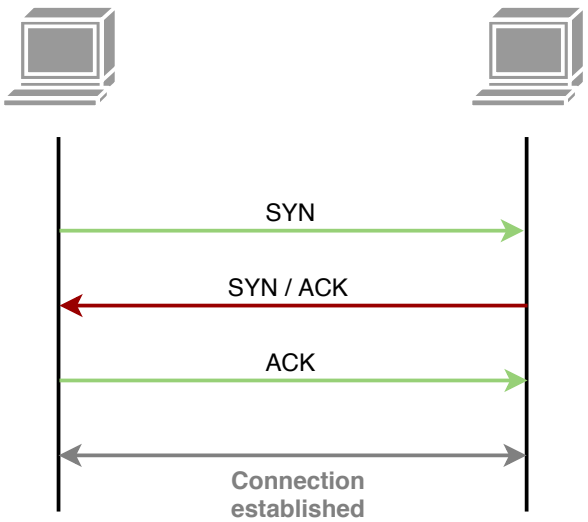

Figure 4.3: Representation of TCP three-way handshake. Only after the 3 steps are completed is that the connection can be established, creating a TCP session between two hosts.

Finally, the packet_count attribute, counts all the packets received and transmitted during the sampling windows time, allowing to distinguish periodic communications and understand

the silence periods, that are computed, posteriorly, inside of each sliding window.

4.2.2.2 Data exiltration case

In data exfiltration scene the concern is now direct to the external network. The sample attributes have to describe the egress flow of data and they are described in Table 4.5.

Table 4.5: Low level description of the attributes defined in each sampling window for the data exfiltration scenario

Sample attributes	Description
ip_external	Number of external IPs contacted
external_bytes_up	Total number of external bytes uploaded
external_bytes_down	Total number of external bytes downloaded
packet_count	Number of packets transmitted and received
tcp_sessions_22	Number of TCP sessions established to port 22
tcp_sessions_80	Number of TCP sessions established to port 80
tcp_sessions_443	Number of TCP sessions established to port 443
tcp_sessions_other	Number of TCP sessions established to other ports

The reason of each one is:

- ip_external

  The number of external IP addresses contacted allows detecting an internal endpoint that initiates a weirded activity by starting to contacting external peers, where its normal behavior is communicating with other inside hosts. For instance, setting up unusual connections to an FTP server placed on the outside network.
- external_bytes_up and external_bytes_down

  The volume of data uploaded/downloaded to the foreign network, likewise in propagation case, depicts the abnormal amount of data send to outside and variations in upload/download peer ration.
- tcp_sessions

  In the leakage data scenario, only SCP, HTTP and HTTPS services are allowed. So, by counting the number of sessions to ports 22, 80, 443, can exhibit an irregular increase of established sessions to an external IP address, where a conversation hike could mean a rise in data sent between peers. On the other hand, by counting the number of TCP sessions to the forbidden ports, illicit irruptions could be pointed. Notice that, to calculate the number TCP sessions it was used the same strategy described on propagation case (4.2.2.2). Yet, in this situation, TCP sessions are defined in function of the port number, once only this application ports had permissions to pass through the enterprise firewall.

4.2.3 Network features formation

Of the first layer of computation (i.e., the counters generation) resulted in 6 and 8 samples for the propagation and the exfiltration case, respectively. Moreover, taking into consideration

the sampling time window assumptions described in Section 3.2.1, it was adopted an interval of 1 second, allowing the detection of periodic events and a faster decision response. Yet, the attributes computed in each sampling window required to be transformed into valuable observations that represented the network activity. In this way, inside of each sliding window was made the feature computation, as described in Section 3.2.2, where *average*, *median*, *variance*, *kurtosis* and *skewness* were calculated over every column, producing 30 and 40 time-independent features, for the propagation and data exfiltration case. Furthermore, also was included the number of silent periods (interval of time defined by the sliding window where the number of packets is zero) and the average duration of these periods. Both of them are linked to network traffic activity, being an added value to distinguish human from machine behavioral and to draw a more reliable network profile (once a host had periods of inactivity). Similar to the sampling window and as discussed in Section 3.2.2, the sliding window dimension should be carefully chosen. In order to obtain a nearer real-time response time without compromising the system's detectability, was empowered a 2 minutes sliding window with a shit time of 5 seconds.

The final feature observations are formed, missing only the categorization of each observation. Being the samples stored in a matrix, the labeling process becomes easy to implement, assigning the known label for each entry. At this moment, data is ready to step into the pipeline and be consumed by classifiers to disclose unnoticed pattern and forming a profile capable of detect patterns deviations.

4.3 Dataset composition

Since there are two distinctive scenarios, each had its dataset. The number of packets, filtered by TCP and UDP protocols, for the propagation case are illustrated in Figure 4.4. It is notorious the high discrepancy between the number of normal packets and the number of packets generated by the attack vector. Although the propagation attack vector has several dynamics, it acts fast, i.e., from the moment that it can access another host, the exploitation process is spontaneous, being the exchange amount of data dependent of the malicious payload carried by the attack.

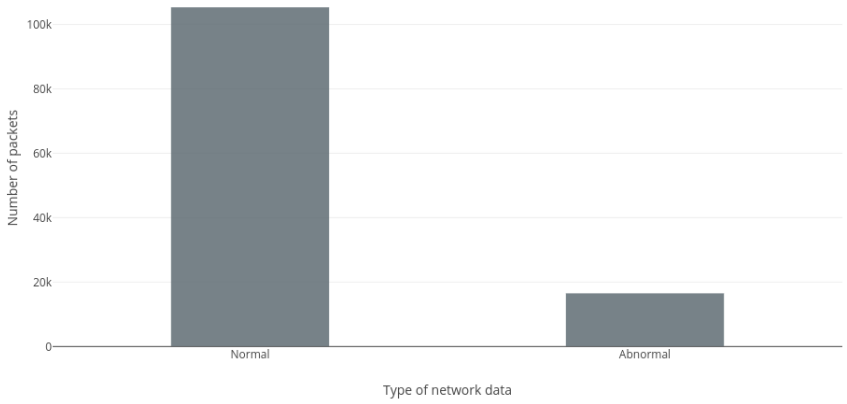

Figure 4.4: Number of normal and abnormal observations that form the propagation's dataset

Regarding the data exfiltration dataset, Figure 4.5 shows the number of packets produced by the three attacks vectors, as well, the normal packets associated with each traffic (HTTP, HTTPS, and SCP). From its analysis, what stands out is the reduced number of packets generated by the HTTP abnormal activity. The imposed limit on the data size that can be sent using the Twiiter API (as discussed in 4.1.2) may be the reason for these reduced number of packets.

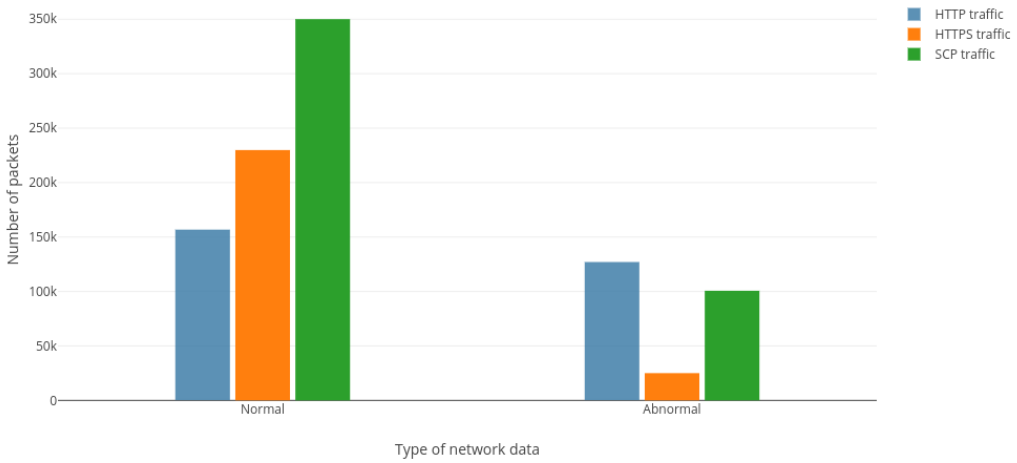

Figure 4.5: Number of normal and abnormal observations relating to the two subsets of data that form the data exfitration scenario.

Following the guidelines described in Section 3.2.4, Figure 4.6 depicts the number of observations resulted after the sliding window strategy, that means, the number of instance per label. However, the data exfiltration scenario is structured by two subsets of network observations:

- SCP dataset (**SCP_D**)

  Defined by file transference normal activity and by SCP attack vector actions, described in Section 4.1.
- HTTP dataset (**HTTP_D**)

  In this case, the data related to normal and abnormal HTTP and HTTPS activities are combined into a single dataset.

The adopted arrangement aims to simplify the complexity of the problem, as discussed in Section 3.3.2, facilitating the ML classifiers action. In such way, the data exfiltration case was broken into tree binary problems constituted by the SCP_D, HTTP_D and a dataset formed by the junction of the last ones, where normal observations are identified with label 0, whereas anomalous observations(SCP_D + HTTP_D) are identified with label 1.

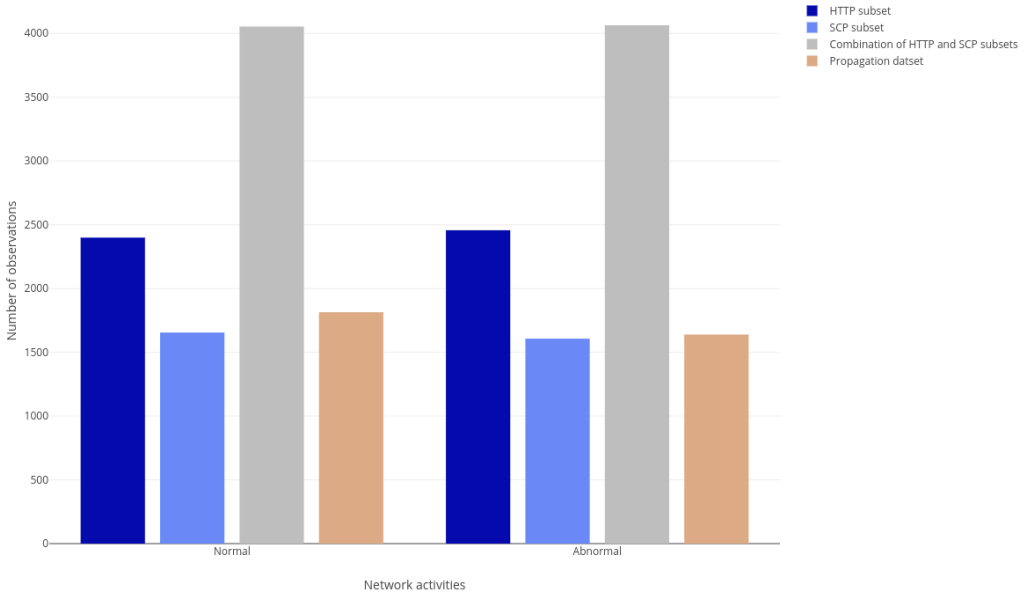

Figure 4.6: Number of network observations generated after the sliding window process, both for the propagation and for the three data exfiltration cases.

From Figure 4.6, it is possible to observe that all datasets are balanced, having a number of normal observations similar to the number of abnormal observations.

According to the guidelines described in Section 3.2.4, another quick way to get an overview of the data that forms a dataset is to plot a histogram. Figure 4.7 shows the representation of some features related to the SCP_D, HTTP_D. From it analyzes, is visible that in both datasets, attributes have different scales, being an indication for the feature scaling process. On the other hand, it is possible to build a pattern comparing the SCP_D and HTTP_D data for the illustrated features. For instance, looking for the SCP_D, it presents a more disperse and frequent distribution in relation to the variance of upload bytes to an external

IP address. Contrarily, for the silence duration, it is visible the absence of periods of silence, concluding that the SCP_D subset exhibit major periods of activity.

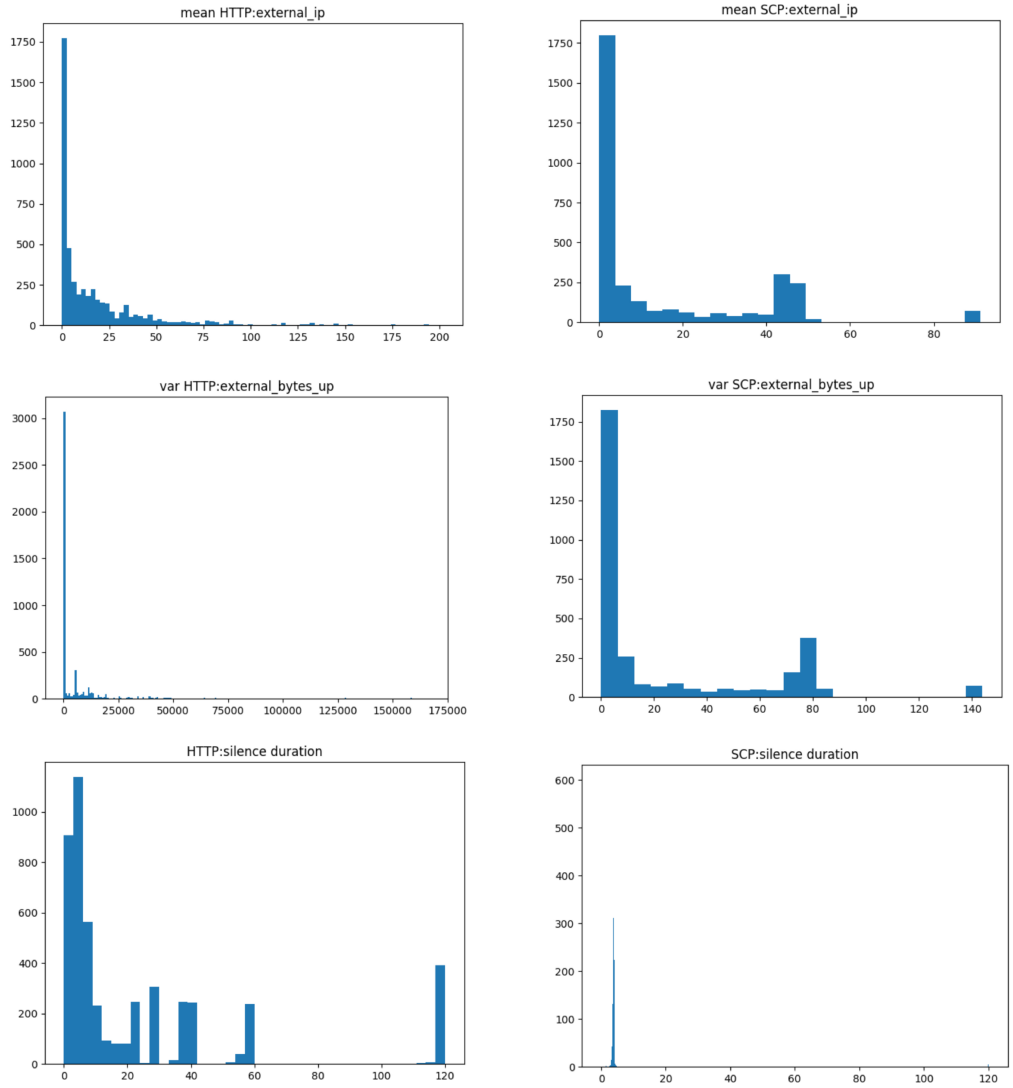

Figure 4.7: Histogram of some features from SCP_D and HTTP_D observations.

An additional approach to have a more concrete perspective of the overall dataset is the PCA analysis, as explained in Section 3.3.1.2. Figure 4.8 and Figure 4.9 plot the cumulative explained variance ratio as a function of the number of components, for the propagation and data exfiltration cases, respectively.

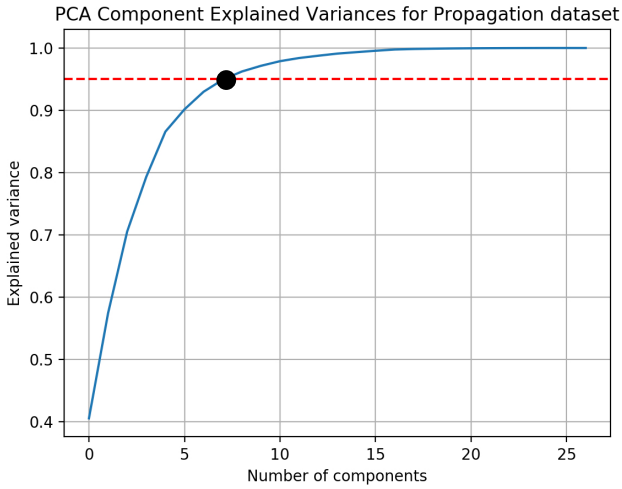

Figure 4.8: Curve that estimate how many components are needed to describe the data over PCA reductions without loss of information, for Propagation dataset

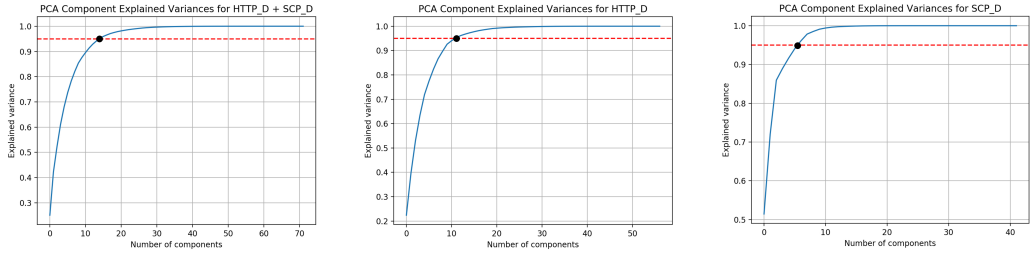

Figure 4.9: Curve that estimate how many components are needed to describe the data over PCA reductions without loss of information for Data exfiltration dataset

Looking into the above figures, the ideal number of features that add up an explained variance of 95% are described in Table 4.6. The ideal number of components resulted from this analysis will be used in the classification methods, Section 4.4. Regarding the HTTP_D it is possible to conclude that applying a two-dimensional projection, a lot of information is lost, exhibiting an explained variance measured under of the 30%.

Table 4.6: Ideal number of components for each dataset when PCA reduction is realized

Dataset	Ideal number of components
Propagation	9
SCP_D	7
HTTP_D	12
SCP_D + HTTP_D	15

4.4 Classification methods and evaluation

In this Proof of Concept it was studied the behavior of SVM, Adaboost, Gradient Boosting, and NN classifiers (described in Section 3.3) over the two existing scenarios. For their

implementation was used the *Scikit-learn* library, where before being put to the test, for each one was obtained the best parameters. Once the parameters are influenced and dependent on the provided data it is essential to tune the model to achieve better results. Yet, for SVM, due to resource and time complexity required to find the optimal parameters, a smooth process was adopted. Regarding the ensemble methods employed (i.e., Adaboost and Gradient Boosting), among the existing parameters, was given special attention to the tree-specific parameter, *max_depth*, and the boosting parameters, *learning_rate*, and *n_estimators* . Once both boosting implementations are based on decision trees, it is necessary the supervision of the tree depth (through the *max_depth* parameter) to control over-fitting. Also, the *learning_rate* parameter exposes the impact of each tree on the final outcome. For instance, a low value will need more trees in the ensemble to fit the training set, but the predictions will usually generalize better being computationally more expensive [126]. Further, it is highly correlated with *n_estimators* (which defines the number of sequential trees to be modeled) being necessary tuned according to the *learning_rate* value. Regarding NN, only was considered one layer on its implementation, not been studied the Deep learning field [189].

Moreover, it is important to highlight that for every model it was used the ideal number of components (as described the computation process in Section 4.3) to perform component reduction, applying PCA reduction.

4.4.1 Propagation scenario

Relatively to the lateral movement case, for an ideal number of features equals to 9, in general, all the classifiers had high accuracies as demonstrated in Table 4.7, being the SVM the worst and Gradient Boosting the best one. The presented results were computed using Stratified 10-fold cross-validation, guaranteeing that evaluation is made using the different combinations of training and classification folds. This is an expected result, once ensemble methods by combining a set of weak learners, normally generalize better than a single classifier, and NN is known to perform well even on complex tasks. On the other hand, even the SVM shows very good accuracies using a linear kernel for training, allowing to infer that data is linearly separable.

Table 4.7: Accuracy results for each classifier algorithm in the propagation scenario.

Classifier	Accuracy
SVM	0.962 ± 0.0215
AdaBoost	0.993 ± 0.0128
Gradient Boosting	0.994 ± 0.0086
Neural Networks	0.993 ± 0.0112

As argued in Section 3.3.3, the main evaluation element that allows interpret the overall performance of a classifier is the confusion matrix. Figure 4.10 represents the confusion matrix for the SVM and Gradient Boosting, respectively.

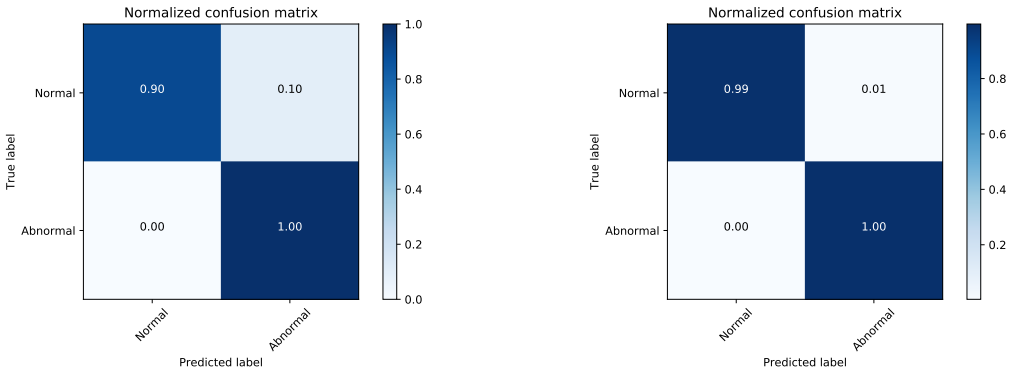

Figure 4.10: Normalized confusion matrix for Support Vector Machine, on the left, and for Gradient Boosting algorithm, on the right side. It displays the miss classifications of the True label to the Predicted label for both Support Vector Machine and Gradient Boosting cases.

Looking for the diagonal elements with darker colors (which represent the correct classifications), the SVM algorithm performs worst, once the Normal label was 36 times classified as Abnormal, meaning the presence of False Positives. While in Gradient Boosting, the error decrease to 1% (only 3 False Positives), having in always a better prediction.

Moreover, it is important to perceive how the model performs in the training and validation data. In this way, Figure 4.11 plots the learning curve for the SVM and Gradient Boosting, showing the model's performance on the training set and the validation set as a function of the training set size.

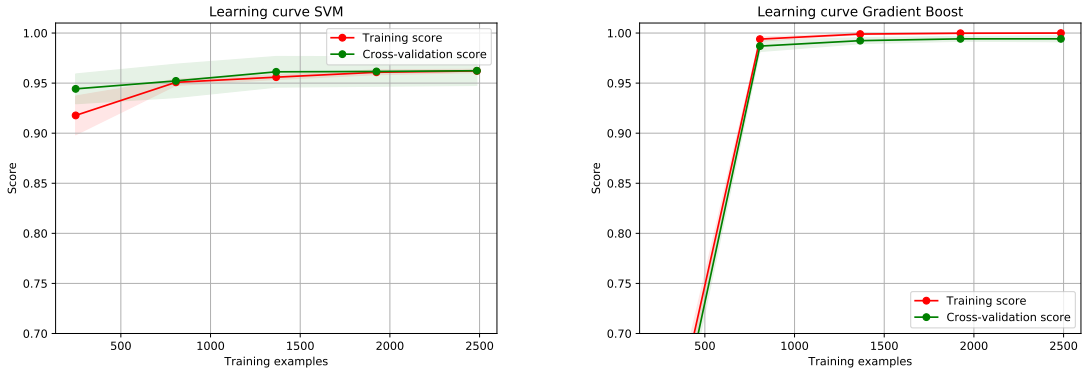

Figure 4.11: Learning curve for the Support Vector Machine and Gradient Boosting classifiers. It shows the validation and training score of an estimator for varying numbers of training samples being essential to understand if models benefit from adding more training data.

From its analysis, the first conclusion is that the Gradient Boosting generalization is superior to the SVM generalization, once the last model reaches 0.95 as the maximum score. Second, about Gradient Boosting, it is possible to infer that the model needs, at least, 800 training samples (approximately) to reach the maximum score and make a better generalization. Unlike the SVM, that both the training and validation scores from the beginning were closed to the maximum. Further, the learning curve can provide a way to verify if the model is over or underfitting by inspecting the gap between the training and validation score. For both

models, that discrepancy it is not visible, once the curves practically match with each other, having a similar performance on training and validation data. At last, since the two scores have already converged to the maximum value, it is valid to conclude that for the provided amount of training examples both models generalize correctly. Yet, in SVM there is space for improvement if more training data was added, in opposition to Gradient Boosting, that more data will not increase the predictor generalization. The SVM upgrade margin can be justified by the superficial search process made for this algorithm parameters, as discussed above in this section.

Additionally, in order to prove the model's generalization in new data, an extra set of data containing observations which are not part of the dataset used to train the model was generated. As explained in Section 4.1.1, Worawit's github was chosen to create the data related to the propagation scenario, however, for the extra data formation was employed the *FuzzBunch* framework that is one of the elements leaked by ShadowBrokers [190]. It can be compared to MetaSploit [191] but written in Python, where provides several ready to use Window's exploits. Using the GNS3 simulation environment, the Eternalblue exploit (using the *FuzzBunch* framework) was launched and spread across the LAN. In this way, the built Gradient Boosting model was applied over the new propagation observations, obtaining an accuracy of 0.839. This result traduces the good generalization capability offered by the trained model over data that it never saw.

4.4.2 Data exfiltration scenario

Being the classifiers objective to discern between anomaly and normality, and as explained in Sections 3.2.4 and 4.3, the data exfiltration scenario was divided into tree sub-cases treated as binary predictions, in order to study the influence of the problem complexity on the performance of the classifiers, by combining the two anomalous data types (SCP_D and HTTP_D) into one. Along these lines, firstly will be analyzed the anomaly detection performance on the individual cases, as the Table 4.8 exposes, being the results calculated using Stratified 10-fold cross-validation.

Table 4.8: Accuracy results obtained from the follow predictors regarding the SCP_D and HTTP_D data.

Classifier	Accuracy for SCP_D	Accuracy for HTTP_D
SVM	0.837 ± 0.044	0.901 ± 0.0243
AdaBoost	0.938 ± 0.0157	0.997 ± 0.0049
Gradient Boosting	0.934 ± 0.0406	0.996 ± 0.0050
Neural Networks	0.868 ± 0.0338	0.989 ± 0.0093

From the above presented Table, once again SVM is the classifier with lower accuracies, as well the ensemble predictors maintained the high performance already registered for the propagation scenario. SVM's accuracy for SCP_D, suggests that SCP observations are not linearly separable, as can be justified by the Figure 4.12.

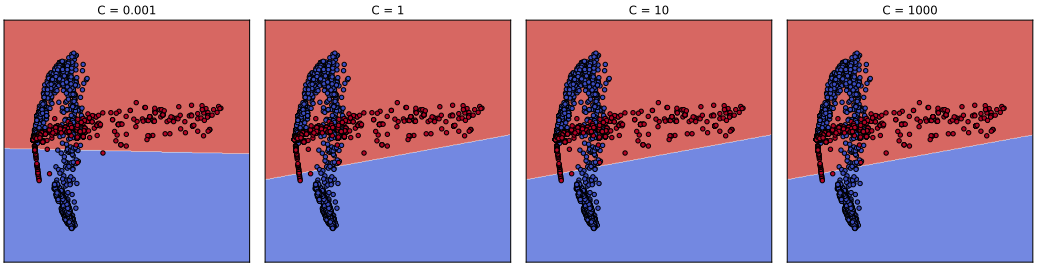

Figure 4.12: Visualization of SVM's C parameter separation margin and the presence of outliers in such partition, showing that SCP_D data is not linearly separable.

Plotting data reducing the features to 2 dimensions using PCA, demonstrating that varying the frontier margins, data is not easily separable, being visible the outliers existence for all margin alternatives. Yet, increasing the number of components, the nature of observations could be more distinguishable, but in the same way, not being totally separable.

Moreover, NN's accuracy results for the SCP_D are unexpected. By interpreting the left plot of Figure 4.13 is visible that the NN algorithm already had converged to the maximum score, not having room to introduce significant improvements.

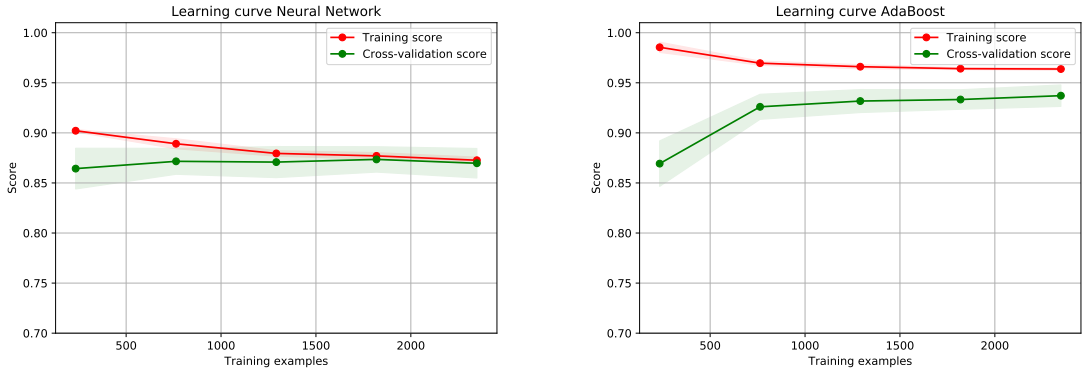

Figure 4.13: Learning curve for the Neural Networks and Adaboost classifiers regarding to the SCP_D data.

These surprising results could be related to the use of only one layer in NN for the training, but a more in-depth analysis is needed to justify this statement. Also, despite Adaboost being the more accurate predictor, in the SCP_D case, it is possible to enhance the performance by including more examples for training the model, as the right learning curve plot illustrates.

Regarding HTTP_D binary classification predicted by the two poor algorithms, if an activity is an irregularity or not, ROC curve for SVM, represented in the left side of Figure 4.14, shows that the true positive rate corner is around 0.15 false positive rate. On the other hand, the right plot displays the NN's ROC curve with a much more smother corner for the optimal true positive rate, followed by a lower 0.1 false positive rate.

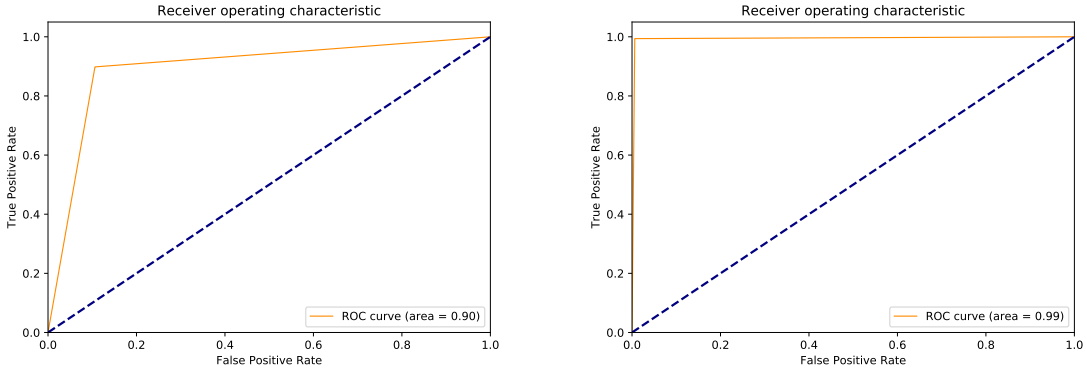

Figure 4.14: ROC curve for Support Vector Machine (on the left) and Neural Networks (on the right) with HTTP_D data, showing the compromise between the true positive and false negative rate.

So, NN grants a more correct prediction of anomalies preserving the low false positive rate, downsizing the normal actions classified as an abnormality, than SVM. This could be sustained by the fact that SVM uses a linear kernel to make the data separation, while NN empowers a more flexible decision function. Also, is remarkable the generalization ability provided by NN, once it was not done an in-depth analysis of all the parameters, demonstrating the potential of this algorithm.

When the SCP_D and HTTP_D are grouped into a single dataset where normal activity has label 0, and both types of anomalies are assigned with label 1, the accuracies results obtained, in general, are better than expected, as Table 4.9 shows, once the problem complexity rise due to the non-differentiation between incursions types.

Table 4.9: Accuracies results obtained by the predictors when the two subsets of the data exfiltration scenario are combine. Note that all results was measured using Stratified 10-fold validation.

Classifier	**Accuracy for SCP_D + HTTP_D**
SVM	0.861 ± 0.0308
AdaBoost	0.972 ± 0.0125
Gradient Boosting	0.971 ± 0.0138
Neural Networks	0.929 ± 0.0240

Comparing the performance results of tables 4.8 and 4.9, Adaboost is the best predictor in the three situations, yet on the SCP_D + HTTP_D data, the difference between Adaboost and Gradient Boosting is insignificant. It is visible the good generalization achieved by Adaboost, but as presented in Figure 4.15, the normalized confusion matrix shows the presence of 0.02% of false positives, that is, 17 normal observations classified as anomalies, and contrarily, 0.04% abnormal instances categorized as normal, resulting in 28 false negatives. In this way, it is necessary to take into account the trade-off between false negatives and false positives to enhance models rehabilitate. In addition, considering the performance results using the Adaboost algorithm following a binary approach adopted in the two scenarios (model is fed with all exfiltration network observations or by one attack vector observations), it is not

conclusive to infer which of the approaches generalized better and prove the initial idea: increased classification difficulty when combined various attack vectors; being necessary to carry out more tests to clear up this situation.

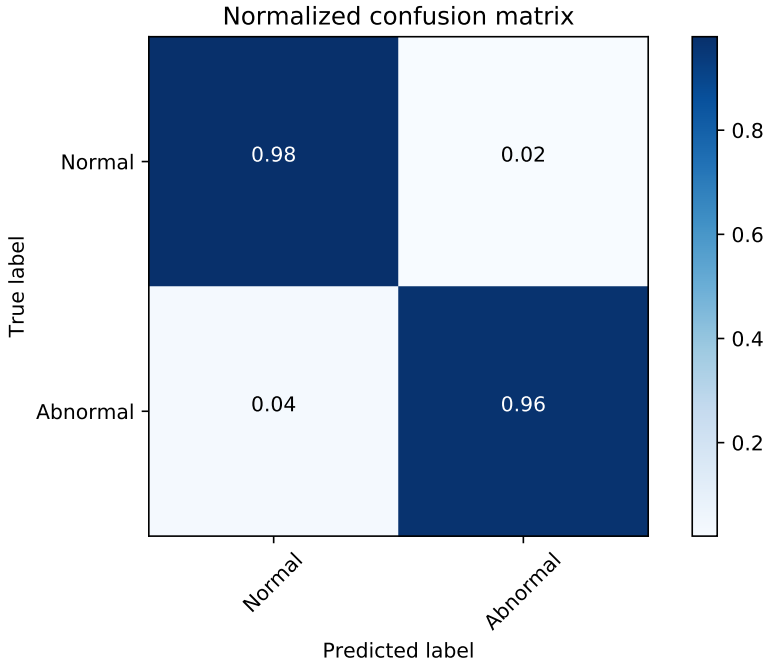

Figure 4.15: Adaabost's confusion matrix for SCP_D + HTTP_D data exhibiting the false and true negatives existence.

4.5 Conclusion

In conclusion, following the guideline detailed in the previous chapter, the results obtained in this Proof of Concept demonstrated to be positive, with space to improve. The created models demonstrated a good generalization capability using the specified features present on the dataset built from numerous captures. However, some results need further research to be interpreted. A real corporate network dataset would made the models more generic and robust.

CHAPTER 5

Conclusions and Future work

Regarding the work done in this dissertation, security communications over Internet and the protection of private assets are tightly linked. Network security topics are an increasingly recurring theme, facing the number of breaches spoiled by companies. New types of tools and mechanism are rising to audit the network welfare, being the monitoring of data needed to improve the anomaly detection process. Looking for the data that flows through the network borders it becomes essential to locate the threat before spreading and causing major damages, mitigating the information disclosed and other severe impacts.

In view of the network data collection, the implementation strategy employed in this work, regarding the used sniffer and its placement, is not the most desirable one, either in terms of application or efficiency. In a real corporation network scenario, an implementation with a port mirror or TAP, as described in section 2.4, is required facing the bulk of data that enterprises deal every day. Other consideration, is the uselessness of the saved captures after network traffic are gathered. This is only for academic purposes since in a production environment, will cause problems related to the resource consumption and their protection (another security concern). However, the analysis procedure over the raw data and, the subsequent document-oriented storage, makes the system more scalable, not being required extra software solutions like databases.

Due to the resource limitation, either at the software or hardware level, it was not possible to create a network environment equal to a true company topology. The empowered proof of concept scenario, described in section 4.1, was designed following the network architecture principles, yet far from the real scenarios.

Concerning the network traffic generation topic performed in this dissertation, although present-day attack vectors have been used to produce network data (likewise the Eternalblue exploit and steganography's techniques applied in social networks), data related to information theft over SCP and propagation cases, were generated on a controlled simulation environment, which may not traduce what really occurs in a enterprise network and influence the ML classification results. Moreover, a more profound study and analysis over creation of the

normal and anomalous actions should be done as future work, to trace a more reliable behavior.

Concerning feature extraction level (4.2.2), the set of features selected for both the propagation and data exfiltration scenarios, proved to be sufficient for describing the network state and to ML models get a good performance in the anomaly classification. All the feature extraction phase were designed to be a general approach and not to cover only the particular empowered attack vectors, being the developed system able to detect other variations of the malicious outbreaks used in the Proof of Concept. Although this solution can be extended for other scenarios and services, both for propagation and data leakage, by taking, for example, other exfiltration methods that employ different network protocols (e.g., DNS, ICMP, FTP and so one) as covert channels. Due to the objective segregation into two distinct cases, i.e., internal propagation and data exfiltration, it was not possible to implement all researched possibilities for each domain, existing space to improve the current solution.

Taking into account the used of ML algorithms to differentiate between anomaly or not, and considering all the took stages until the model performance evaluation, as described in section 3.3, in general, for the two scenarios, the ensemble algorithms demonstrated a better performance compared with the remaining. Supported by confusion matrix analyses, Gradient Boosting is the model which better guarantees the compromise between having high detectability and a small number of false positives, regrading to propagation's context (section 4.4.1). Beside that, this model proves its generalization competence being challenged with new network observations, created by a variation of the attack vector used for training, that clearly distance from those which compose the original dataset. For the data exfiltration scenario (section 4.4.2) assisted by ROC curve and models accuracies, models also exposed satisfactory results, being Adaboost algorithm the best predictor. Despite the good accuracies and generalization obtained, it would be necessary test ML models with real company network observations for a finer evaluation. Also, new benchmarks containing new ML algorithms should be considered as future work. Additionally, it will be interesting take advantage of the NN potential by adopting it, in order to evaluate the Deep Learning performance in the anomaly classification.

In order to improve the network environment conceived, considering as future work other equipments which are part of a company, for instance, printers, smart TVs, smartphones and other IoT devices; instead of dealing only with host machines, will lead to creating a more robust system.Under these additional network elements, the quantity and variety of traffic that travels across network rise, and in this way, it's possible to collect more significant data and extrapolate learned patterns to future ones.

In favor of enhancing and making the feature extraction process more robust, other elements should be recognized. Firstly, instead of only considering the number of established TCP sessions, accounting also the re-connections, the failed attempts and the flag ration for the data stream in question. Secondly, correlate the volume, number of packets, TCP/UDP sessions, etc, to a specific IP or port could give a more detailed and low-level overview of the network. Furthermore, for a better understanding of the silence periods distribution, the variance of these inactivity periods should be included. Another improvement to be considered

as future work is the inclusion of wavelets to examine the actions frequency, allowing the models to cover periodicity aspects in the network patterns definition. Yet, it is always important to have in mind that increasing the number of attributes will rise the final number of features consumed by ML algorithms.

Furthermore, it would be advantageous the application of counter-measures after the detection of an anomaly. Tracing back the irregularity origins, identifying the compromised machines and isolating them from the rest of the network by applying network segregation techniques or more drastically, block the IP address of the infected machines in the firewall, it is a future work that would have high applicability in a real company. Complementary, instead of just being considered the use of ML algorithms for anomaly detection, an additional possible approach is to benefit from these new-fashioned methodologies to classify network services, as some literature suggested [192]–[194]. The identification of HTTP, FTP, DNS services, or even more ambitious, the recognition of encrypted data (for example, VoIP and VPN traffic), by traffic observation could be extremely fruitful to extended network intrusion detection strategies. However, privacy legal aspects rise with this type of deep inspection, as debated in section 2.1.6.

All in all, companies spend absurd amounts of money to strengthen their security but, increasingly are exposed to data breaches [195]. The current solutions that empower ML capabilities are rising and overpower the existing traditional tools (some described in section 2.6), being necessary to emphasize that security breaches can be minimized, can be mitigated, yet cannot be extinguished. Many legitimate security firms today offer services that can, at very least, protect against well-known vulnerabilities, drawing products as if it were the best and could combat all kinds of threats, leaving corporations susceptible. So, is this lack of security that does huge damages to corporations caused by the absence of necessary means or provoked by the "snake oil" products [196] sold by security vendors just to increase their profits?

References

[1] Clearswift, *Insider Threat: 74% of security incidents come from the extended enterprise, not hacking groups | Clearswift*, 2017. [Online]. Available: https://www.clearswift.com/about-us/pr/press-releases/insider-threat-74-security-incidents-come-extended-enterprise-not-hacking-groups (visited on 02/24/2018).

[2] I. Winkler and A. Treu Gomes, *Advanced Persistent Security: A Cyberwarfare Approach to Implementing Adaptive Enterprise Protection, Detection, and Reaction Strategies*, eng. Amsterdam, Boston, Heidelberg: Elsevier, 2017, ISBN: 978-0-12-809316-0.

[3] *CIA Triad*. [Online]. Available: http://resources.infosecinstitute.com/cia-triad/ (visited on 02/21/2018).

[4] J. Andress and S. Winterfeld, *The basics of information security: understanding the fundamentals of InfoSec in theory and practice*, eng, 2. ed, ser. The basics. Amsterdam: Elsevier/Syngress, 2014, OCLC: 911327935, ISBN: 978-0-12-800744-0.

[5] S. Qadir and S. M. K. Quadri, "Information Availability: An Insight into the Most Important Attribute of Information Security", *Journal of Information Security*, vol. 07, no. 03, pp. 185–194, 2016, ISSN: 2153-1234, 2153-1242. DOI: 10.4236/jis.2016.73014. [Online]. Available: http://www.scirp.org/journal/doi.aspx?DOI=10.4236/jis.2016.73014 (visited on 02/21/2018).

[6] A. M. d. S. P. d. Moraes, *Cisco firewalls: concepts, design and deployment for Cisco stateful firewall solutions*, eng, 1. print, ser. Cisco Press networking technology series Security. Indianapolis, Ind: Cisco Press, 2011, OCLC: 837977809, ISBN: 978-1-58714-109-6.

[7] Y. Wang, J. Wei, and K. Vangury, "Bring your own device security issues and challenges", in *Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th*, IEEE, 2014, pp. 80–85, ISBN: 978-1-4799-2355-7. DOI: 10.1109/CCNC.2014.6866552.

[8] W. M. Stout and V. E. Urias, "Challenges to securing the Internet of Things", in *Security Technology (ICCST), 2016 IEEE International Carnahan Conference on*, IEEE, 2016, pp. 1–8. DOI: 10.1109/CCST.2016.7815675.

[9] C. Paque, *Network Security Concepts and Policies > Building Blocks of Information Security*, 2013. [Online]. Available: http://www.ciscopress.com/articles/article.asp?p=1998559 (visited on 02/22/2018).

[10] N. Saper, "International cryptography regulation and the global information economy", *Nw. J. Tech. & Intell. Prop.*, vol. 11, p. xv, 2012.

[11] A. Nogueira and P. Salavador, *A Pratical Approach to Corpoarte Networks Engeneering*, en. River Publishers, 2013, ISBN: 978-87-92982-09-4.

[12] C. N. Academy, *Connecting Networks Companion Guide*. Pearson Education, 2014.

[13] J. Tiso, *Designing Cisco Network Service Architectures (ARCH)*, 3rd ed. Cisco Press, 2011, ISBN: 978-1-58714-288-8.

[14] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, *Network Traffic Anomaly Detection and Prevention*, ser. Computer Communications and Networks. Cham: Springer International Publishing, 2017, ISBN: 978-3-319-65186-6 978-3-319-65188-0. DOI: 10.1007/978-3-319-65188-0. [Online]. Available: http://link.springer.com/10.1007/978-3-319-65188-0 (visited on 02/21/2018).

[15] Cisco, “Zone-Based Policy Firewall Design and Application Guide”, *Cisco*, p. 59, 2010.

[16] J. Frahim, O. Santos, and A. Ossipov, *Cisco ASA: All-in-one Next-Generation Firewall, IPS, and VPN Services*, 3rd ed. Cisco Press, 2014, ISBN: 978-1-58714-307-6.

[17] J. M. Kizza, *Guide to Computer Network Security*, ser. Computer Communications and Networks. Cham: Springer, 2017, ISBN: 978-3-319-55605-5 978-3-319-55606-2. DOI: 10.1007/978-3-319-55606-2. [Online]. Available: http://link.springer.com/10.1007/978-3-319-55606-2 (visited on 02/21/2018).

[18] S. Jose, “Cisco Unified Communications Manager Managed Services Guide, Release 8.6(1)”, *Cisco*, p. 956, 2011.

[19] Juniper Networks, *Understanding RADIUS Accounting - Technical Documentation - Support - Juniper Networks*, 2016. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/radius-accounting-understanding-qfx-series.html (visited on 03/16/2018).

[20] D. Chapman, S. Cooper, and E. Zwicky, *Building Internet Firewalls*, Second. O’Reilly Media, 2000, ISBN: 1-56592-871-7.

[21] U. Troppens, U. Troppens, and U. Troppens, Eds., *Storage networks explained: basics and application of Fibre Channel SAN, NAS, iSCSI, InfiniBand and FCoE*, eng, 2nd ed. Chichester, West Sussex, U.K: Wiley, 2009, OCLC: ocn318641425, ISBN: 978-0-470-74143-6.

[22] IBM Redbooks, *Introduction to Storage Area Networks and System Networking*, EN, 9th ed. Vervante, 2017, ISBN: 978-0-7384-3713-2.

[23] Microsoft Corporation, *Microsoft SMB Protocol and CIFS Protocol Overview (Windows)*. [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/aa365233(v=vs.85).aspx (visited on 06/21/2018).

[24] Z. Huili, “Realization of Files Sharing between Linux and Windows Based on Samba”, in *Future BioMedical Information Engineering, 2008. FBIE’08. International Seminar on*, IEEE, Dec. 2008, pp. 418–420. DOI: 10.1109/FBIE.2008.97.

[25] Samba, *Samba - opening windows to a wider world.* [Online]. Available: https://www.samba.org/ (visited on 06/21/2018).

[26] Mozilla, *HTTP*, en-US, 2018. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP (visited on 07/16/2018).

[27] Cisco, *AAA Protocols*, en, 2016. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/net_mgmt/cisco_secure_access_control_system/5-2/user/guide/acsuserguide/rad_tac_phase.html (visited on 03/19/2018).

[28] Juniper Networks, *Understanding Central Network Access Using RADIUS and TACACS+ - Technical Documentation - Support - Juniper Networks*, 2018. [Online]. Available: https://www.juniper.net/documentation/en_US/junos-space-apps/network-director2.5/topics/concept/radius-tacacs-understanding.html (visited on 03/19/2018).

[29] Cisco, *How Does RADIUS Work?*, en, 2006. [Online]. Available: https://www.cisco.com/c/en/us/support/docs/security-vpn/remote-authentication-dial-user-service-radius/12433-32.html (visited on 03/19/2018).

[30] ——, “Configuring IEEE 802.1x Port-Based Authentication”, in *Catalyst 3750-X and 3560-X Switch Software Configuration Guide,* Cisco, 2017.

[31] ——, *Authentication on Wireless LAN Controllers Configuration Examples*, en, 2010. [Online]. Available: https://www.cisco.com/c/en/us/support/docs/wireless-mobility/wlan-security/82135-wlc-authenticate.html (visited on 03/19/2018).

[32] ——, “MAC Authentication Bypass Deployment Guide”, p. 23, 2011.

[33] D. Deshmukh and B. Iyer, “Design of IPSec virtual private network for remote access”, in *Computing, Communication and Automation (ICCCA), 2017 International Conference on*, IEEE, May 2017, pp. 716–719, ISBN: 978-1-5090-6471-7. DOI: 10.1109/CCAA.2017.8229894. [Online]. Available: http://ieeexplore.ieee.org/document/8229894/ (visited on 03/20/2018).

[34] Check Point, *Check Point Stateful Inspection Technology*. [Online]. Available: https://www.checkpoint.com/smb/help/utm1/8.0/7080.htm (visited on 03/22/2018).

[35] Juniper Networks, "Junos Network Secure", p. 2, 2015.

[36] F. Hock and P. Kortiš, "Commercial and open-source based Intrusion Detection System and Intrusion Prevention System (IDS/IPS) design for an IP networks", in *Emerging eLearning Technologies and Applications (ICETA), 2015 13th International Conference on*, IEEE, 2015, pp. 1–4.

[37] R. Samrin and D. Vasumathi, "Review on anomaly based network intrusion detection system", in *Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), 2017 International Conference on*, IEEE, 2017, pp. 141–147.

[38] M. A. Rahaman, C. Hebert, and J. Frank, "An attack pattern framework for monitoring enterprise information systems", in *Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2016 IEEE 25th International Conference on*, IEEE, 2016, pp. 173–178.

[39] S. M. Alqahtani and R. John, "A Comparative Analysis of Different Classification Techniques for Cloud Intrusion Detection Systems' Alerts and Fuzzy Classifiers", in *Computing Conference, 2017*, IEEE, 2017, pp. 406–415, ISBN: 978-1-5090-5443-5. DOI: 10.1109/SAI.2017.8252132.

[40] C. Schon, N. Adams, and M. Evangelou, "Clustering and monitoring edge behaviour in enterprise network traffic", in *Intelligence and Security Informatics (ISI), 2017 IEEE International Conference on*, IEEE, 2017, pp. 31–36.

[41] A. Pal Singh and M. Deep Singh, "Analysis of Host-Based and Network-Based Intrusion Detection System", *International Journal of Computer Network and Information Security*, vol. 6, no. 8, pp. 41–47, Jul. 2014, ISSN: 20749090, 20749104. DOI: 10.5815/ijcnis.2014.08.06. [Online]. Available: http://www.mecs-press.org/ijcnis/ijcnis-v6-n8/v6n8-6.html (visited on 02/21/2018).

[42] N. N. Mkuzangwe and F. Nelwamondo, "Ensemble of classifiers based network intrusion detection system performance bound", in *Systems and Informatics (ICSAI), 2017 4th International Conference on*, IEEE, 2017, pp. 970–974.

[43] J. Kevric, S. Jukic, and A. Subasi, "An effective combining classifier approach using tree algorithms for network intrusion detection", en, *Neural Computing and Applications*, vol. 28, no. S1, pp. 1051–1058, Dec. 2017, ISSN: 0941-0643, 1433-3058. DOI: 10.1007/s00521-016-2418-1. [Online]. Available: http://link.springer.com/10.1007/s00521-016-2418-1 (visited on 02/21/2018).

[44] R. M. Yousufi, P. Lalwani, and M. Potdar, "A network-based intrusion detection and prevention system with multi-mode counteractions", in *Innovations in Information, Embedded and Communication Systems (ICIIECS), 2017 International Conference on*, IEEE, 2017, pp. 1–6.

[45] D. Stiawan, A. H. Abdullah, and M. Y. Idris, "The trends of intrusion prevention system network", in *Education Technology and Computer (ICETC), 2010 2nd International Conference on*, vol. 4, IEEE, 2010, pp. V4–217.

[46] A. Chaudhary and A. Sardana, "Software Based Implementation Methodologies for Deep Packet Inspection", in *Information Science and Applications (ICISA), 2011 International Conference on*, IEEE, Apr. 2011, pp. 1–10. DOI: 10.1109/ICISA.2011.5772430.

[47] A. Saurabh, *A Guide to Deep Packet Inspection*, en-US, Jul. 2017. [Online]. Available: http://blog.catchpoint.com/2017/07/19/guide-deep-packet-inspection/ (visited on 03/06/2018).

[48] *EUR-Lex - 32002l0058 - EN*, EN, text/html; charset=UNICODE-1-1-UTF-8. [Online]. Available: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32002L0058:EN:HTML (visited on 03/08/2018).

[49] T. Margoni and M. Perry, "Legal Consequences of Packet Inspection", en, *SSRN Electronic Journal*, 2011, ISSN: 1556-5068. DOI: 10.2139/ssrn.2028981. [Online]. Available: http://www.ssrn.com/abstract=2028981 (visited on 03/20/2018).

[50] G. Greenwald and E. MacAskill, *NSA Prism program taps in to user data of Apple, Google and others*, en, Jun. 2013. [Online]. Available: http://www.theguardian.com/world/2013/jun/06/us-tech-giants-nsa-data (visited on 03/23/2018).

[51] E. Council Forbes Technology, *How One Security Setting Can Solve The Web Encryption Problem*, en, 2017. [Online]. Available: https://www.forbes.com/sites/forbestechcouncil/2017/05/18/how-one-security-setting-can-solve-the-web-encryption-problem/ (visited on 03/23/2018).

[52] R. Gaddam and M. Nandhini, "An analysis of various snort based techniques to detect and prevent intrusions in networks proposal with code refactoring snort tool in Kali Linux environment", in *Inventive Communication and Computational Technologies (ICICCT), 2017 International Conference on*, IEEE, 2017, pp. 10–15, ISBN: 978-1-5090-5297-4. DOI: 10.1109/ICICCT.2017.7975177.

[53] E. Gandotra, D. Bansal, and S. Sofat, "Zero-day malware detection", in *Embedded Computing and System Design (ISED), 2016 Sixth International Symposium on*, IEEE, 2016, pp. 171–175.

[54] D. Last, "Forecasting zero-day vulnerabilities", in *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, ACM, 2016, p. 13, ISBN: 978-1-4503-3752-6. DOI: 10.1145/2897795.2897813. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2897795.2897813 (visited on 02/23/2018).

[55] X. Sun, J. Dai, P. Liu, A. Singhal, and J. Yen, "Towards probabilistic identification of zero-day attack paths", in *Communications and Network Security (CNS), 2016 IEEE Conference on*, IEEE, 2016, pp. 64–72.

[56] *CVE - Common Vulnerabilities and Exposures (CVE)*. [Online]. Available: https://cve.mitre.org/ (visited on 02/28/2018).

[57] E. Gandotra, D. Bansal, and S. Sofat, "Malware analysis and classification: A survey", *Journal of Information Security*, vol. 5, no. 02, p. 56, 2014.

[58] *What is Data Exfiltration?*, Text, Apr. 2015. [Online]. Available: https://digitalguardian.com/blog/what-data-exfiltration (visited on 02/26/2018).

[59] T. Yadav and A. M. Rao, "Technical aspects of cyber kill chain", in *International Symposium on Security in Computing and Communication*, Springer, 2015, pp. 438–452.

[60] J. R. C. Nurse, O. Buckley, P. A. Legg, M. Goldsmith, S. Creese, G. R. T. Wright, and M. Whitty, "Understanding Insider Threat: A Framework for Characterising Attacks", IEEE, May 2014, pp. 214–228, ISBN: 978-1-4799-5103-1. DOI: 10.1109/SPW.2014.38. [Online]. Available: http://ieeexplore.ieee.org/document/6957307/ (visited on 02/24/2018).

[61] L. Xiangyu, L. Qiuyang, and S. Chandel, "Social engineering and insider threats", in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2017 International Conference on*, IEEE, 2017, pp. 25–34, ISBN: 978-1-5386-2209-4. DOI: 10.1109/CyberC.2017.91. [Online]. Available: http://ieeexplore.ieee.org/document/8250331/ (visited on 02/24/2018).

[62] D. L. Costa, M. L. Collins, S. J. Perl, M. J. Albrethsen, G. J. Silowash, and D. L. Spooner, "An ontology for insider threat indicators development and applications", CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, Tech. Rep., 2014.

[63] M. Collins, "Common sense guide to mitigating insider threats", CARNEGIE-MELLON UNIV PITTSBURGH PA PITTSBURGH United States, Tech. Rep., 2016.

[64] "Profile: Edward Snowden", en-GB, *BBC News*, Dec. 2013. [Online]. Available: http://www.bbc.com/news/world-us-canada-22837100 (visited on 02/25/2018).

[65] L. Liu, O. De Vel, Q.-L. Han, J. Zhang, and Y. Xiang, "Detecting and Preventing Cyber Insider Threats: A Survey", *IEEE Communications Surveys & Tutorials*, pp. 1–1, 2018, ISSN: 1553-877X. DOI: 10.1109/COMST.2018.2800740. [Online]. Available: http://ieeexplore.ieee.org/document/8278157/ (visited on 02/24/2018).

[66] F. L. Greitzer, J. R. Strozer, S. Cohen, A. P. Moore, D. Mundie, and J. Cowley, "Analysis of Unintentional Insider Threats Deriving from Social Engineering Exploits", IEEE, May 2014, pp. 236–250, ISBN: 978-1-4799-5103-1. DOI: 10.1109/SPW.2014.39. [Online]. Available: http://ieeexplore.ieee.org/document/6957309/ (visited on 02/24/2018).

[67] Ponemon Institute, "2016 Cost of Insider Threats Benchmark Study of Organizations in the United States", Tech. Rep., 2016, p. 20.

[68] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research", in *Insider Attack and Cyber Security*, Springer, 2008, pp. 69–90.

[69] F. Ullah, M. Edwards, R. Ramdhany, R. Chitchyan, M. A. Babar, and A. Rashid, "Data exfiltration: A review of external attack vectors and countermeasures", en, *Journal of Network and Computer Applications*, vol. 101, pp. 18–54, Jan. 2018, ISSN: 10848045. DOI: 10.1016/j.jnca.2017.10.016. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804517303569 (visited on 02/26/2018).

[70] B. G. Schlicher, L. P. MacIntyre, and R. K. Abercrombie, "Towards reducing the data exfiltration surface for the insider threat", in *System Sciences (HICSS), 2016 49th Hawaii International Conference on*, IEEE, 2016, pp. 2749–2758, ISBN: 978-0-7695-5670-3. DOI: 10.1109/HICSS.2016.345. [Online]. Available: http://ieeexplore.ieee.org/document/7427528/ (visited on 02/26/2018).

[71] S. Fiegerman, *The biggest data breaches ever*, Sep. 2017. [Online]. Available: http://money.cnn.com/2017/09/07/technology/business/biggest-breaches-ever/index.html (visited on 03/23/2018).

[72] A. V.-B. Pelisson Anaele, *The Equifax hack isn't the biggest security breach of all time, but it could be one of the worst in history for Americans*, 2017. [Online]. Available: http://www.businessinsider.com/how-equifax-compares-to-biggest-hacks-of-all-time-chart-2017-9 (visited on 03/23/2018).

[73] X. Shu, K. Tian, and A. Ciambrone, "Breaking the target: An analysis of target data breach and lessons learned", *arXiv preprint arXiv:1701.04940*, 2017.

[74] A. Peterson, "The Sony Pictures hack, explained", en-US, *Washington Post*, Dec. 2014, ISSN: 0190-8286. [Online]. Available: https://www.washingtonpost.com/news/the-switch/wp/2014/12/18/the-sony-pictures-hack-explained/ (visited on 03/26/2018).

[75] A. Greenberg, *HBO Hackers Drop Ransom Note and More Game of Thrones Spoilers | WIRED*, 2017. [Online]. Available: https://www.wired.com/story/hbo-hack-ransom-note/ (visited on 03/26/2018).

[76] E. Brumaghin, *Ransom Where? Malicious Cryptocurrency Miners Takeover, Generating Millions*, 2018. [Online]. Available: http://blog.talosintelligence.com/2018/01/malicious-xmr-mining.html (visited on 03/12/2018).

[77] J. Bloomberg, *Top Cyberthreat Of 2018: Illicit Cryptomining*, 2018. [Online]. Available: https://www.forbes.com/sites/jasonbloomberg/2018/03/04/top-cyberthreat-of-2018-illicit-cryptomining/#4f26ebff5ae8 (visited on 03/12/2018).

[78] M. H. Bhuyan, H. J. Kashyap, D. K. Bhattacharyya, and J. K. Kalita, "Detecting distributed denial of service attacks: Methods, tools and future directions", *The Computer Journal*, vol. 57, no. 4, pp. 537–556, 2013.

[79] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, and M. Kallitsis, "Understanding the mirai botnet", in *USENIX Security Symposium*, 2017.

[80] A. C. Estes, *The Era of Chaos-Inducing Ransomware Is Here and It's Scary as Hell*, en-US, 2017. [Online]. Available: https://gizmodo.com/the-era-of-chaos-inducing-ransomware-is-here-and-its-sc-1796485449 (visited on 07/06/2018).

[81] L. Abrams, *WannaCry / Wana Decryptor / WanaCrypt0r Info & Technical Nose Dive*, 2017. [Online]. Available: https://www.bleepingcomputer.com/news/security/wannacry-wana-decryptor-wanacrypt0r-info-and-technical-nose-dive/ (visited on 07/06/2018).

[82] A. Chiu, *Player 3 Has Entered the Game: Say Hello to 'WannaCry'*, 2017. [Online]. Available: http://blog.talosintelligence.com/2017/05/wannacry.html (visited on 07/06/2018).

[83] BBC, *Cyber-attack: Europol says it was unprecedented in scale - BBC News*, 2017. [Online]. Available: http://www.bbc.com/news/world-europe-39907965 (visited on 03/26/2018).

[84] BetaFred, *Microsoft Security Bulletin MS17-010 - Critical*, en-us, 2017. [Online]. Available: https://docs.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010 (visited on 03/26/2018).

[85] C. Cimpanu, *Bad Rabbit Ransomware Outbreak Hits Eastern Europe*, 2017. [Online]. Available: https://www.bleepingcomputer.com/news/security/bad-rabbit-ransomware-outbreak-hits-eastern-europe/ (visited on 03/26/2018).

[86] A. Hern, *WannaCry, Petya, NotPetya: How ransomware hit the big time in 2017*, en, Dec. 2017. [Online]. Available: http://www.theguardian.com/technology/2017/dec/30/wannacry-petya-notpetya-ransomware (visited on 03/26/2018).

[87] L. C. o. March 8 and 2018, *Worm Infects Redis, Windows Servers with Cryptomining Malware*, en-US, Mar. 2018. [Online]. Available: https://securityboulevard.com/2018/03/worm-infects-redis-and-windows-servers-with-cryptomining-malware/ (visited on 03/26/2018).

[88] J. C. Wong and O. Solon, *Massive ransomware cyber-attack hits nearly 100 countries around the world*, en, May 2017. [Online]. Available: http://www.theguardian.com/technology/2017/may/12/global-cyber-attack-ransomware-nsa-uk-nhs (visited on 03/26/2018).

[89] A. Chiu, *New Ransomware Variant "Nyetya" Compromises Systems Worldwide*, 2017. [Online]. Available: http://blog.talosintelligence.com/2017/06/worldwide-ransomware-variant.html (visited on 07/07/2018).

[90] C. Cimpanu, *Petya Ransomware Outbreak Originated in Ukraine via Tainted Accounting Software*, en-us, 2017. [Online]. Available: https://www.bleepingcomputer.com/news/security/petya-ransomware-outbreak-originated-in-ukraine-via-tainted-accounting-software/ (visited on 07/07/2018).

[91] K. Zetter, *Inside the Cunning, Unprecedented Hack of Ukraine's Power Grid*, Mar. 2016. [Online]. Available: https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/ (visited on 06/25/2018).

[92] C. Francescani, *How Chinese-language social media may help fuel Chinese-American conservatism*, en, Apr. 2016. [Online]. Available: https://www.nbcnews.com/news/asian-america/how-chinese-language-social-media-may-help-fuel-chinese-american-n883386 (visited on 06/25/2018).

[93] J. Markoff, "Stuxnet Software Worm Hit 5 Industrial Facilities in Iran", en-US, *The New York Times*, Feb. 2011, ISSN: 0362-4331. [Online]. Available: https://www.nytimes.com/2011/02/13/science/13stuxnet.html (visited on 06/25/2018).

[94] C. Stoneff, *The Seven Steps of a Successful Cyber Attack*, en-US, Jun. 2015. [Online]. Available: https://resources.infosecinstitute.com/the-seven-steps-of-a-successful-cyber-attack/ (visited on 06/25/2018).

[95] P. Engebretson, *The basics of hacking and penetration testing: ethical hacking and penetration testing made easy*. Elsevier, 2013.

[96] K. Krombholz, H. Hobel, M. Huber, and E. Weippl, "Advanced social engineering attacks", en, *Journal of Information Security and Applications*, vol. 22, pp. 113–122, Jun. 2015, ISSN: 22142126. DOI: 10.1016/j.jisa.2014.09.005. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S2214212614001343 (visited on 02/27/2018).

[97] N. Y. Conteh and P. J. Schmick, "Cybersecurity:risks, vulnerabilities and countermeasures to prevent social engineering attacks", *International Journal of Advanced Computer Research*, vol. 6, no. 23, pp. 31–38, Feb. 2016, ISSN: 22497277, 22777970. DOI: 10.19101/IJACR.2016.623006. [Online]. Available: http://accentsjournals.org/PaperDirectory/Journal/IJACR/2016/3/1.pdf (visited on 02/27/2018).

[98] B. Gardner and V. Thomas, *Building an Information Security Awareness Program: Defending Against Social Engineering and Technical Threats*, 1st ed. Syngress, 2014, ISBN: 978-0-12-419967-5.

[99] D. Perera, *Researcher: Sony hackers used fake emails*, 2015. [Online]. Available: https://www.politico.com/story/2015/04/sony-hackers-fake-emails-117200.html (visited on 03/27/2018).

[100] C. Plett and L. Poggemeye, *Netstat*, en-us, 2017. [Online]. Available: https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/netstat (visited on 03/28/2018).

[101] *Nmap: The Network Mapper - Free Security Scanner*. [Online]. Available: https://nmap.org/ (visited on 03/28/2018).

[102] A. Bohara, M. A. Noureddine, A. Fawaz, and W. H. Sanders, "An unsupervised multi-detector approach for identifying malicious lateral movement", in *Reliable Distributed Systems (SRDS), 2017 IEEE 36th Symposium on*, IEEE, 2017, pp. 224–233, ISBN: 978-1-5386-1679-6. DOI: 10.1109/SRDS.2017.31. [Online]. Available: http://ieeexplore.ieee.org/document/8069085/ (visited on 03/28/2018).

[103] TrendMicro, "Lateral Movement: How Do Threat Actors Move Deeper Into Your Network?", en, p. 7, 2013.

[104] B. Schneier, *The Story Behind The Stuxnet Virus*, 2010. [Online]. Available: https://www.forbes.com/2010/10/06/iran-nuclear-computer-technology-security-stuxnet-worm.html#22d8b5ff51e8 (visited on 03/29/2018).

[105] D. Kushner, "The real story of stuxnet", *ieee Spectrum*, vol. 3, no. 50, pp. 48–53, 2013.

[106] Ryan C. Van Antwerp, "Exfiltration techniques: An examination and emulation", PhD thesis, University of Delaware, 2011.

[107] A. A. Attaby, M. F. Mursi Ahmed, and A. K. Alsammak, "Data hiding inside JPEG images with high resistance to steganalysis using a novel technique: DCT-M3", en, *Ain Shams Engineering Journal*, Feb. 2017, ISSN: 20904479. DOI: 10.1016/j.asej.2017.02.003. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S209044791730031X (visited on 03/29/2018).

[108] C. Wueest, "Targeted Attacks Against the Energy Sector", en, *Symatec*, p. 29, 2014.

[109] Symantec, "Attacks on point-of-sales systems", Symantec, Tech. Rep., 2014.

[110] M. Collins, *Network security through data analysis: building situational awareness*, en, 1. ed. Beijing: O'Reilly, 2014, OCLC: 862760366, ISBN: 978-1-4493-5790-0.

[111] Jacques du Toit, *Active vs. Passive network monitoring: An infographic*, en-US, 2016. [Online]. Available: undefined (visited on 07/01/2018).

[112] A. Cecil, *A Summary of Network Traffic Monitoring and Analysis Techniques*. [Online]. Available: https://www.cse.wustl.edu/~jain/cse567-06/ftp/net_monitoring/index.html (visited on 07/02/2018).

[113] iPerf, *iPerf - The TCP, UDP and SCTP network bandwidth measurement tool*. [Online]. Available: https://iperf.fr/ (visited on 07/02/2018).

[114] Juniper Networks, *Understanding Port Mirroring - Technical Documentation - Support - Juniper Networks*, 2018. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/port-mirroring-qfx-series-understanding.html (visited on 07/02/2018).

[115] Gigamon, "White Paper: TAP vs SPAN", en, p. 4, 2017.

[116] A. Daly, "The legality of deep packet inspection", en, p. 12, 2010.

[117] Let's Encrypt, *Let's Encrypt Stats - Let's Encrypt - Free SSL/TLS Certificates*, 2018. [Online]. Available: https://letsencrypt.org/stats/#growth (visited on 07/03/2018).

[118] K. Finley, *Half the Web Is Now Encrypted. That Makes Everyone Safer | WIRED*, 2017. [Online]. Available: https://www.wired.com/2017/01/half-web-now-encrypted-makes-everyone-safer/ (visited on 07/03/2018).

[119] N. T. Van, T. N. Thinh, and L. T. Sach, "An anomaly-based network intrusion detection system using deep learning", in *System Science and Engineering (ICSSE), 2017 International Conference on*, IEEE, 2017, pp. 210–214, ISBN: 978-1-5386-3422-6. DOI: 10.1109/ICSSE.2017.8030867. [Online]. Available: http://ieeexplore.ieee.org/document/8030867/ (visited on 04/10/2018).

[120] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges", en, *Computers & Security*, vol. 28, no. 1-2, pp. 18–28, Feb. 2009, ISSN: 01674048. DOI: 10.1016/j.cose.2008.08.003. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167404808000692 (visited on 04/09/2018).

[121] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks.", in *Lisa*, vol. 99, 1999, pp. 229–238.

[122] V. Jyothsna, V. R. Prasad, and K. M. Prasad, "A review of anomaly based intrusion detection systems", *International Journal of Computer Applications*, vol. 28, no. 7, pp. 26–35, 2011.

[123] K. Surrey and R. Pyke, *Detecting hackers (analyzing network traffic) by poisson model measure*, 2015.

[124] B. Marr, *The Top 10 AI And Machine Learning Use Cases Everyone Should Know About*, en, 2016. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2016/09/30/what-are-the-top-10-use-cases-for-machine-learning-and-ai/ (visited on 04/02/2018).

[125] *Computer Technology Helps Radiologists Spot Overlooked Small Breast Cancers | Cancer Network*, 2000. [Online]. Available: http://www.cancernetwork.com/articles/computer-technology-helps-radiologists-spot-overlooked-small-breast-cancers (visited on 04/02/2018).

[126] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, en, 1st ed. O'Reilly Media, 2014, ISBN: 978-1-4919-6229-9.

[127] T. Mitchell, *Machine Learning*. 1997, ISBN: 0-07-042807-7.

[128] B. Marr, *What Is The Difference Between Artificial Intelligence And Machine Learning?*, 2016. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2016/12/06/what-is-the-difference-between-artificial-intelligence-and-machine-learning/#27a1b3122742 (visited on 04/02/2018).

[129] S. Marsland, *Machine Learning: An Algorithmic Perspective*, en, 2nd ed. Chapman and Hall/CRC, 2014, ISBN: 978-1-4665-8328-3.

[130] M. Hauskrecht, I. Batal, M. Valko, S. Visweswaran, G. F. Cooper, and G. Clermont, "Outlier Detection for Patient Monitoring and Alerting", *Journal of biomedical informatics*, vol. 46, no. 1, pp. 47–55, Feb. 2013, ISSN: 1532-0464. DOI: 10.1016/j.jbi.2012.08.004. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3567774/ (visited on 04/03/2018).

[131] S. Kaski and J. Peltonen, "Dimensionality Reduction for Data Visualization [Applications Corner]", *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 100–104, Mar. 2011, ISSN: 1053-5888. DOI: 10.1109/MSP.2010.940003.

[132] S. Knapton, *AlphaGo Zero: Google DeepMind supercomputer learns 3,000 years of human knowledge in 40 days*, 2017. [Online]. Available: https://www.telegraph.co.uk/science/2017/10/18/alphago-zero-google-deepmind-supercomputer-learns-3000-years/ (visited on 04/03/2018).

[133] M. Banko and E. Brill, "Scaling to Very Very Large Corpora for Natural Language Disambiguation", in *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, Toulouse, France: Association for Computational Linguistics, Jul. 2001, pp. 26–33. DOI: 10.3115/1073012.1073017. [Online]. Available: http://www.aclweb.org/anthology/P01-1005 (visited on 04/05/2018).

[134] A. Halevy, P. Norvig, and F. Pereira, "The Unreasonable Effectiveness of Data", *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 8–12, Mar. 2009, ISSN: 1541-1672. DOI: 10.1109/MIS.2009.36.

[135] Quora, *How Is Big Data Changing The Business Landscape?*, en, 2018. [Online]. Available: https://www.forbes.com/sites/quora/2018/03/12/how-is-big-data-changing-the-business-landscape/ (visited on 06/26/2018).

[136] The Oxford Math Center, *Famous Statistical Blunders in History*, en, Oxford College. [Online]. Available: /drupal7/node/251 (visited on 04/05/2018).

[137] J. Grus, *Data science from Scratch: first principles with Python*, en, 1. ed. Beijing: O'Reilly, 2015, OCLC: 912307506, ISBN: 978-1-4919-0142-7.

[138] D. H. Wolpert, "The Lack of A Priori Distinctions Between Learning Algorithms", en, *Neural Computation*, vol. 8, no. 7, pp. 1341–1390, Oct. 1996, ISSN: 0899-7667, 1530-888X. DOI: 10.1162/neco.1996.8.7.1341. [Online]. Available: http://www.mitpressjournals.org/doi/10.1162/neco.1996.8.7.1341 (visited on 09/09/2018).

[139] A. Dainotti, A. Pescape, and G. Ventre, "Worm Traffic Analysis and Characterization", en, IEEE, Jun. 2007, pp. 1435–1442, ISBN: 978-1-4244-0353-0. DOI: 10.1109/ICC.2007.241. [Online]. Available: http://ieeexplore.ieee.org/document/4288912/ (visited on 08/08/2018).

[140] Universita' degli Studi di Napoli "Federico II", *..:: Traffic and Tools ::..* [Online]. Available: http://www.grid.unina.it/Traffic/ (visited on 08/08/2018).

[141] B. S. Everitt and A. Skrondal, *The Cambridge Dictionary of Statistics*, en, 4th ed. Cambridge University Press, 2010.

[142] D. Veitch and P. Abry, "A wavelet-based joint estimator of the parameters of long-range dependence", en, *IEEE Transactions on Information Theory*, vol. 45, no. 3, pp. 878–897, Apr. 1999, ISSN: 00189448. DOI: 10.1109/18.761330. [Online]. Available: http://ieeexplore.ieee.org/document/761330/ (visited on 08/08/2018).

[143] A. Wagner and B. Plattner, "Entropy based worm and anomaly detection in fast IP networks", in *14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE'05)*, Jun. 2005, pp. 172–177. DOI: 10.1109/WETICE.2005.35.

[144] C. E. Shannon, "A mathematical theory of communication", *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[145] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An Overview of IP Flow-Based Intrusion Detection", *IEEE Communications Surveys Tutorials*, vol. 12, no. 3, pp. 343–356, 2010, ISSN: 1553-877X. DOI: 10.1109/SURV.2010.032210.00054.

[146] A. Altaher, S. Ramadass, and A. Almomani, "Real time network anomaly detection using relative entropy", in *8th International Conference on High-capacity Optical Networks and Emerging Technologies*, Dec. 2011, pp. 258–260. DOI: 10.1109/HONET.2011.6149829.

[147] D. Yao, M. Yin, J. Luo, and S. Zhang, "Network Anomaly Detection Using Random Forests and Entropy of Traffic Features", in *2012 Fourth International Conference on Multimedia Information Networking and Security*, Nov. 2012, pp. 926–929. DOI: 10.1109/MINES.2012.146.

[148] N. Sarnsuwan, C. Charnsripinyo, and N. Wattanapongsakorn, "A new approach for internet worm detection and classification", in *Networked Computing (INC), 2010 6th International Conference on*, IEEE, 2010, pp. 1–4.

[149] V. S. Koganti, L. K. Galla, and N. Nuthalapati, "Internet worms and its detection", in *2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Dec. 2016, pp. 64–73. DOI: 10.1109/ICCICCT.2016.7987920.

[150] K. Limthong and T. Tawsook, "Network traffic anomaly detection using machine learning approaches", in *2012 IEEE Network Operations and Management Symposium*, Apr. 2012, pp. 542–545. DOI: 10.1109/NOMS.2012.6211951.

[151] K. Limthong, "Real-Time Computer Network Anomaly Detection Using Machine Learning Techniques", en, *Journal of Advances in Computer Networks*, pp. 1–5, 2013, ISSN: 17938244. DOI: 10.7763/JACN.2013.V1.1. [Online]. Available: http://www.jacn.net/show-7-9-1.html (visited on 08/09/2018).

[152] A. Al-Bataineh and G. White, "Analysis and detection of malicious data exfiltration in web traffic", en, in *2012 7th International Conference on Malicious and Unwanted Software*, IEEE, Oct. 2012, pp. 26–31, ISBN: 978-1-4673-4879-9 978-1-4673-4880-5 978-1-4673-4878-2. DOI: 10.1109/MALWARE.2012.6461004. [Online]. Available: http://ieeexplore.ieee.org/document/6461004/ (visited on 08/10/2018).

[153] I. Ahmed and K.-s. Lhee, "Classification of packet contents for malware detection", en, *Journal in Computer Virology*, vol. 7, no. 4, pp. 279–295, Nov. 2011, ISSN: 1772-9890, 1772-9904. DOI: 10.1007/s11416-011-0156-6. [Online]. Available: http://link.springer.com/10.1007/s11416-011-0156-6 (visited on 08/13/2018).

[154] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update", en, *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, p. 10, Nov. 2009, ISSN: 19310145. DOI: 10.1145/1656274.1656278. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1656274.1656278 (visited on 08/13/2018).

[155] Y. Liu, C. Corbett, K. Chiang, R. Archibald, B. Mukherjee, and D. Ghosal, "SIDD: A Framework for Detecting Sensitive Data Exfiltration by an Insider Attack", in *2009 42nd Hawaii International Conference on System Sciences*, Jan. 2009, pp. 1–10. DOI: 10.1109/HICSS.2009.390.

[156] C. S. Burru, R. A. Gopinath, and H. Gu, "Wavelets and Wavelet Transforms", en, *Upper Saddle River*, p. 291, 1998.

[157] R. Ramachandran, S. Neelakantan, and A. S. Bidyarthy, "Behavior model for detecting data exfiltration in network environment", in *2011 IEEE 5th International Conference on Internet Multimedia Systems Architecture and Application*, Dec. 2011, pp. 1–5. DOI: 10.1109/IMSAA.2011.6156340.

[158] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction", in *European conference on computer vision*, Springer, 2000, pp. 751–767.

[159] P. Sharma, A. Joshi, and T. Finin, "Detecting data exfiltration by integrating information across layers", in *Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on*, IEEE, 2013, pp. 309–316. [Online]. Available: http://ieeexplore.ieee.org/document/6642487/.

[160] *Snort - Network Intrusion Detection & Prevention System*, 2018. [Online]. Available: https://www.snort.org/ (visited on 07/04/2018).

[161] *The Bro Network Security Monitor*, 2018. [Online]. Available: https://www.bro.org/ (visited on 07/04/2018).

[162] *Open Information Security Foundation | Community Driven, Open Source*, 2017. [Online]. Available: https://oisf.net/ (visited on 07/04/2018).

[163] Cisco, "Cisco Stealthwatch - Improving visibility across your business", Cisco, Tech. Rep., 2017.

[164] Darktrace, *Darktrace | Products*, 2018. [Online]. Available: https://www.darktrace.com/products/ (visited on 07/05/2018).

[165] N. Kobie, *Darktrace's AI is now automatically responding to hacks – and stopping them*, 2017. [Online]. Available: http://www.wired.co.uk/article/darktrace-machine-learning-security (visited on 07/05/2018).

[166] S. F. Leal, M. Rosario Oliveira, and R. Valadas, "Anomaly detection of Internet traffic using robust feature selection based on kernel density estimation", en, in *2015 European Conference on Networks and Communications (EuCNC)*, Paris, France: IEEE, Jun. 2015, pp. 482–486, ISBN: 978-1-4673-7359-3. DOI: 10.1109/EuCNC.2015.7194122. [Online]. Available: http://ieeexplore.ieee.org/document/7194122/ (visited on 09/09/2018).

[167] E. Alpaydin, *Introduction to Machine Learning*, Second. MIT Press, 2010, ISBN: 978-0-262-01243-0.

[168] scikit-learn, *Kernel PCA — scikit-learn 0.19.2 documentation*, 2017. [Online]. Available: http://scikit-learn.org/stable/auto_examples/decomposition/plot_kernel_pca.html#sphx-glr-auto-examples-decomposition-plot-kernel-pca-py (visited on 09/10/2018).

[169] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution", en, *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001. DOI: 10.1162/089976601750264965. [Online]. Available: http://www.mitpressjournals.org/doi/10.1162/089976601750264965 (visited on 09/07/2018).

[170] Y. Wang, J. Wong, and A. Miner, "Anomaly intrusion detection using one class svm", in *Information assurance workshop*, 2004, pp. 358–364. [Online]. Available: http://ieeexplore.ieee.org/document/1437839/.

[171] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs decision trees in intrusion detection systems", in *Proceedings of the 2004 ACM symposium on Applied computing*, ACM, 2004, pp. 420–424.

[172] M. Kumar, M. Hanumanthappa, and T. V. S. Kumar, "Intrusion Detection System using decision tree algorithm", in *2012 IEEE 14th International Conference on Communication Technology*, Nov. 2012, pp. 629–634. DOI: 10.1109/ICCT.2012.6511281.

[173] KDnuggets, *XGBoost, a Top Machine Learning Method on Kaggle, Explained*, en-US. [Online]. Available: https://www.kdnuggets.com/2017/10/xgboost-top-machine-learning-method-kaggle-explained.html,%20https://www.kdnuggets.com/2017/10/xgboost-top-machine-learning-method-kaggle-explained.html (visited on 08/28/2018).

[174] Z. Xue-qin, G. Chun-hua, and L. Jia-jun, "Intrusion detection system based on feature selection and support vector machine", in *Communications and Networking in China, 2006. ChinaCom'06. First International Conference on*, IEEE, 2006, pp. 1–5.

[175] G. Poojitha, K. N. Kumar, and P. J. Reddy, "Intrusion Detection using Artificial Neural Network", in *Computing Communication and Networking Technologies (ICCCNT), 2010 International Conference on*, IEEE, Jul. 2010, pp. 1–7. DOI: 10.1109/ICCCNT.2010.5592568.

[176] J. Shun and H. A. Malki, "Network Intrusion Detection System Using Neural Networks", in *Natural Computation, 2008. ICNC'08. Fourth International Conference on*, IEEE, vol. 5, Oct. 2008, pp. 242–246. DOI: 10.1109/ICNC.2008.900.

[177] scikit-learn, *One-class SVM with non-linear kernel (RBF) — scikit-learn 0.19.2 documentation*, 2018. [Online]. Available: http://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html (visited on 09/07/2018).

[178] ——, *Receiver Operating Characteristic (ROC) — scikit-learn 0.19.2 documentation*, 2017. [Online]. Available: http://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html (visited on 08/06/2018).

[179] Y. Raviv, *EternalBlue - Everything There Is To Know*, en-US, Sep. 2017. [Online]. Available: http://research.checkpoint.com/eternalblue-everything-know/ (visited on 07/06/2018).

[180] worawit, *Ms17-010*, https://github.com/worawit/MS17-010, 2018.

[181] C. Cimpanu, *Exploit Derived From ETERNALSYNERGY Upgraded to Target Newer Windows Versions*, 2017. [Online]. Available: https://www.bleepingcomputer.com/news/security/exploit-derived-from-eternalsynergy-upgraded-to-target-newer-windows-versions/ (visited on 07/07/2018).

[182] Rapid7, *CVE-2017-0143 MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution | Rapid7*. [Online]. Available: https://www.rapid7.com/db/modules/exploit/windows/smb/ms17_010_psexec (visited on 07/07/2018).

[183] *Python http requests for humans*, https://github.com/requests/requests, 2018.

[184] J. Bardin, *Scp module for paramiko*, https://github.com/jbardin/scp.py, 2018.

[185] Kaspersky Lab, *Kaspersky Lab Identifies Worrying Trend in Hackers Using Steganography | Kaspersky Lab US*, 2017. [Online]. Available: https://usa.kaspersky.com/about/press-releases/2017_kaspersky-lab-identifies-worrying-trend-in-hackers-using-steganography (visited on 07/09/2018).

[186] R. McGrath, *Actively maintained, pure python wrapper for the twitter api. supports both normal and streaming twitter apis.* https://github.com/ryanmcgrath/twython, 2018.

[187] *Kaitai struct: Declarative language to generate binary data parsers*, https://github.com/kaitai-io/kaitai_struct, 2018.

[188] C. A. Lucas, *Python network packet dissection frameworks shootout: Scapy vs Construct vs Hachoir vs Kaitai Struct*, en, Mar. 2017. [Online]. Available: https://pythonistac.wordpress.com/2017/03/09/python-network-packet-dissection-frameworks-shootout-scapy-vs-construct-vs-hachoir-vs-kaitai-struct/ (visited on 07/17/2018).

[189] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", vol. 521, no. 7553, p. 436, May 2015, ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature14539. [Online]. Available: http://www.nature.com/articles/nature14539 (visited on 09/05/2018).

[190] Door Rik van Duijn, *Playing around with NSA's hacking tools*, nl, Apr. 2017. [Online]. Available: https://www.dearbytes.com/blog/playing-around-with-nsa-hacking-tools/ (visited on 09/10/2018).

[191] *Metasploit | Penetration Testing Software, Pen Testing Security*, en. [Online]. Available: https://www.metasploit.com/ (visited on 09/10/2018).

[192] M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, and M. Saberian, "Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning", *arXiv:1709.02656 [cs]*, Sep. 2017, arXiv: 1709.02656. [Online]. Available: http://arxiv.org/abs/1709.02656 (visited on 09/12/2018).

[193] Z. Wang, "The applications of deep learning on traffic identification", *BlackHat USA*, 2015.

[194] R. Alshammari and A. N. Zincir-Heywood, "Machine learning based encrypted traffic classification: Identifying SSH and Skype", en, in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Ottawa, ON, Canada: IEEE, Jul. 2009, pp. 1–8, ISBN: 978-1-4244-3763-4. DOI: 10.1109/CISDA.2009.5356534. [Online]. Available: http://ieeexplore.ieee.org/document/5356534/ (visited on 09/12/2018).

[195] John Leyden, *More data lost or stolen in first half of 2017 than the whole of last year*, en, 2017. [Online]. Available: https://www.theregister.co.uk/2017/09/20/gemalto_breach_index/ (visited on 09/12/2018).

[196] D. Cameron, *Snake Oil Salesmen Plague the Security Industry, But Not Everyone Is Staying Quiet*, en-US, 2018. [Online]. Available: https://gizmodo.com/snake-oil-salesmen-plague-the-security-industry-but-no-1822590687 (visited on 09/12/2018).